A multidimensional method-of-lines transport scheme for atmospheric flows over steep terrain using arbitrary meshes

James Shaw[a,*], Hilary Weller[a], John Methven[a], Terry Davies[b]

[a]*Department of Meteorology, University of Reading, Reading, United Kingdom*
[b]*Met Office, Exeter, United Kingdom*

Abstract

The inclusion of terrain in atmospheric models gives rise to mesh distortions near the lower boundary that can degrade accuracy and challenge the stability of transport schemes. Multidimensional transport schemes avoid all splitting errors on distorted and arbitrarily structured meshes, and Eulerian schemes have low computational cost, relying entirely on reconstructions at fixed points.

This paper presents a multidimensional Eulerian finite volume transport scheme, "cubicFit", which is designed to be numerically stable on arbitrary meshes and which is second-order convergent regardless of mesh distortions. Constraints derived from a von Neumann stability analysis are imposed during model initialisation to obtain stability and improve accuracy in distorted regions of the mesh, and near steeply-sloping lower boundaries. The reconstruction method has a low computational cost because most calculations depend upon the mesh only, with just one vector multiply per face needed per time-step.

The cubicFit scheme is evaluated using two, idealised numerical tests of atmospheric flow. The first is a new test case that assesses accuracy near the ground by transporting a tracer over steep terrain on severely distorted terrain-following meshes and cut-cell meshes. The second is a standard test case that deforms a tracer in a vortical flow on hexagonal icosahedra and cubed-sphere meshes. In all tests, cubicFit is stable and largely insensitive to mesh distortions, and cubicFit results are more accurate than those obtained using a multidimensional linear upwind transport scheme.

Keywords: Finite volume, unstructured mesh, atmospheric modelling, least-squares approximation, von Neumann stability analysis

1. Introduction

Numerical simulations of atmospheric flows solve equations of motion that result in the transport of momentum, temperature, water species and trace gases. The numerical representation of Earth's terrain complicates the transport problem because the mesh is necessarily distorted by the modification of the lower boundary. As new atmospheric models use increasingly fine mesh spacing, meshes are able to resolve steep, small-scale slopes. Numerical schemes in operational weather forecast models can perform poorly over large mountain ranges, exhibiting small-scale numerical noise in momentum [1], temperature, humidity [2] and potential vorticity fields [3], or even violating the Courant–Friedrich–Lewy stability constraint resulting in so-called 'grid-point storms' [4]. Hence, a transport scheme is desired that yields stable and accurate solutions, particularly near the surface where the weather affects us directly. We present a new transport scheme which is numerically stable on arbitrary meshes and which is generally insensitive to mesh distortions created by steep slopes. It has a low computational cost since most calculations are not repeated every time-step because they depend upon the mesh geometry only.

There are two main methods for representing terrain in atmospheric models: terrain-following layers and cut cells. Both methods modify regular quadrilateral meshes to produce distorted meshes with cells that are aligned in vertical columns. Most operational models use terrain-following layers in which horizontal mesh surfaces are moved

*Corresponding author
Email address: js102@zepler.net (James Shaw)

upwards to accommodate the terrain. A vertical decay function is chosen so that mesh surfaces slope less steeply with increasing height. The most straightforward is the linear decay function used by the basic terrain-following transform [5] (also called the sigma coordinate), but many atmospheric models suffer from large numerical errors on such meshes [2, 6, 7]. To reduce such errors, more complex decay functions have been developed so that mesh surfaces are smoother [8, 2, 9, 6].

An alternative to terrain-following layers is the cut cell method. Cut cell meshes are constructed by 'cutting' a regular quadrilateral mesh with a piecewise-linear representation of the terrain. New vertices are created where the terrain intersects mesh edges, and cell volumes that lie beneath the ground are removed. Cut cell meshes can have arbitrarily small cells that impose severe timestep constraints on explicit transport schemes. Several techniques have been developed to allieviate this problem, known as the 'small-cell problem': small cells can be merged with adjacent cells [10], cell volumes can be artificially increased [11], or an implicit scheme can be used near the ground with an explicit scheme used aloft [12].

Another method for avoiding the small-cell problem was proposed by [13] in which cell vertices are moved vertically so that they are positioned at the terrain surface. In this paper the method is referred to as the slanted cell method in order to distinguish it from the traditional cut cell method. Slanted cell meshes do not suffer from arbitrarily small cells because the horizontal cell dimensions are not modified by the presence of terrain. TODO: *we might improve upon this technique by merging triangular cells*

Smoothed terrain-following layers, cut cells and slanted cell methods all reduce the amount of mesh distortion but the presence of sloping terrain means that any mesh must necessarily be distorted, at least near the ground. Even when distortions are minimal, transport across mesh surfaces tends to be more common near steep slopes, and this misalignment between the flow and mesh surfaces increases numerical errors [14, 2, 13]. A huge variety of transport schemes have been developed for atmospheric models, but few are able to account for distortions associated with steep terrain because they treat horizontal and vertical transport separately TODO: *citations*, resulting in numerical errors called 'splitting errors'. Such errors can be reduced by explicitly accounting for transverse fluxes when combining fluxes [15], but splitting errors are nevertheless apparent in flows over steep terrain where meshes are highly distorted and metric terms in a terrain-following coordinate transform are large [16].

Transport schemes are often classified as dimensionally-split or multidimensional. Dimensionally-split schemes such as [17, 18] calculate transport in each dimension separately before the flux contributions are combined. Such schemes are computationally efficient and allow existing one-dimensional high-order methods to be used. To use a dimensionally-split scheme over terrain, a terrain-following coordinate transform is required. Perhaps confusingly, dimensionally-split schemes are sometimes called multidimensional, too, because they use one-dimensional techniques for multidimensional transport.

Unlike dimensionally-split schemes, multidimensional schemes consider transport in two or three dimensions together. There are several subclasses of multidimensional schemes that include 2D semi-Lagrangian finite volume schemes (also called conservative mesh remapping), swept-area schemes (also called flux-form semi-Lagrangian, incremental remapping, or forward-in-time), and method-of-lines schemes (also called Eulerian schemes). 2D semi-Lagrangian finite volume schemes such as [19, 20] integrate over departure cells that are found by tracing backward the trajectories of cell vertices. These schemes are conservative because departure cells are constructed so that there are no overlaps or gaps, which requires that cell areas are simply-connected domains [21]. Swept area schemes such as [22, 23, 24, 25] calculate the flux through a cell face by integrating over the upstream area that is swept out over one time-step. Such schemes differ in their choice of area approximation, sub-grid reconstruction, and spatial integration method. Because swept area schemes integrate over the reconstructed field, they typically require a matrix-vector multiply per face [25, 23]. Method-of-lines schemes such as [26, 27] use a spatial discretisation to reduce the transport PDE to an ODE that is typically solved using a multi-stage timestepping method. Unlike 2D semi-Lagrangian finite volume schemes, swept area and method-of-lines schemes achieve conservation for non-simply connected domains that can result from small-scale rotational flows [24]. There are many more types of atmospheric transport schemes, but all can be classified according to their treatment of the three spatial dimensions. A more comprehensive overview is presented in [28].

For transport schemes that are ordinarily classified as 'multidimensional', a further distinction ought to made between horizontally-multidimensional and three-dimensional schemes. Multidimensional schemes are almost always horizontally-multidimensional because, while the two horizontal dimensions are considered together, horizontal and

vertical transport are still treated separately. Very few three-dimensional schemes have been suggested for use in atmospheric models [e.g. 29, 30, 31] although such schemes might be expected to be more accurate on distorted meshes with steep terrain. The multidimensional scheme developed in [32] is unusual because it has no horizontal–vertical splitting and it has been used in two-dimensional flows on Cartesian x–z planes with distorted meshes [13, 16].

In this paper, we present a new multidimensional method-of-lines scheme, 'cubicFit', that improves upon the scheme in [32] and avoids all splitting errors. To reconstruct face values, the scheme fits a multidimensional polynomial over a cubic, upwind-biased stencil using a least-squares approach. The implementation uses constraints derived from a von Neumann stability analysis to select appropriate polynomial fits for stencils in highly-distorted mesh regions. The least-squares procedure depends upon the mesh geometry only, hence the scheme has a low computational cost that is comparable to dimensionally-split schemes, requiring only n multiplies per cell face per time-step where n is the size of the stencil.

The remainder of this paper is organised as follows. Section 2 discretises the transport equation using a method-of-lines approach, and provides a complete description of the cubicFit transport scheme and a standard multidimensional linear upwind transport scheme. Section 3 evaluates the cubicFit scheme using two TODO: *three* idealised numerical tests. TODO: *include horizontal advection first* The first is a new two-dimensional test case designed to challenge the stability and accuracy of transport schemes next to a mountainous lower boundary. In this test, a tracer placed at the ground is transported over steep slopes using terrain-following, cut cell and slanted cell meshes. The second is a standard test of deformational flow on a spherical Earth [33], which we use to assess the cubicFit transport scheme on hexagonal icosahedra and cubed-sphere meshes. Concluding remarks are made in section 4.

2. Transport schemes for arbitrary meshes

The transport of a dependent variable ϕ in a prescribed wind field $\mathbf{u}$ is given by the equation

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\mathbf{u}\phi) = 0 \tag{1}$$

In the mass continuity equation, $\phi = \rho$ where ρ is the air density. In the continuity equation for tracer density $\phi = \rho q$ where q is the tracer mixing ratio. For the special case of a non-divergent velocity field then ρ is constant hence $\phi = q$. The time derivative is discretised using a two-stage, second-order Runge-Kutta method, TODO: *we should use a two-stage, second-order Heun scheme, not the mid-point scheme described below*

$$\phi^{\star} = \phi^{(n)} + \frac{\Delta t}{2} f(\phi^{(n)}) \tag{2a}$$

$$\phi^{(n+1)} = \phi^{(n)} + \Delta t f(\phi^{\star}) \tag{2b}$$

where $f(\phi^{(n)}) = -\nabla \cdot (\mathbf{u}\phi^{(n)})$ at time level n. The same time-stepping method is used for both cubicFit and linearUpwind schemes.

Using the finite volume method, the wind field is prescribed at face centroids and the dependent variable is stored at cell centroids. The divergence term in equation (1) is discretised using Gauss's theorem:

$$\nabla \cdot (\mathbf{u}\phi) \approx \frac{1}{\mathcal{V}_c} \sum_{f \in c} \mathbf{u}_f \cdot \mathbf{S}_f \phi_F \tag{3}$$

where subscript f denotes a value stored at a face and subscript F denotes a value approximated at a face from surrounding values stored at cell centres. $\mathcal{V}_c$ is the cell volume, $\mathbf{u}_f$ is a velocity vector prescribed at a face, $\mathbf{S}_f$ is the surface area vector with a direction outward normal to the face and a magnitude equal to the face area, ϕ_F is an approximation of the dependent variable at the face, and $\sum_{f \in c}$ denotes a summation over all faces f bordering cell c. Note that equation (3) is a second-order approximation of the divergence term that limits the cubicFit transport scheme to second-order accuracy.

This discretisation is applicable to arbitrary meshes. A neccesary condition for stability is given by the multidimensional Courant number,

$$\mathrm{Co}_c = \frac{\Delta t}{2\mathcal{V}_c} \sum_{f \in c} \mathbf{u} \cdot \mathbf{S}_f \tag{4}$$

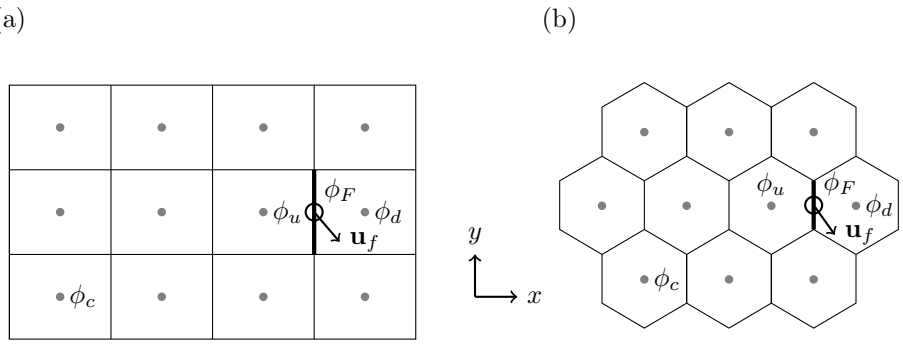

Figure 1: Upwind-biased stencils for faces far away from the boundaries of two-dimensional (a) rectangular and (b) hexagon meshes. The stencil is used to fit a multidimensional polynomial to cell centre values, ϕ_c, marked by grey circles, in order to approximate the value ϕ_F at the face centroid marked by an open circle. ϕ_u and ϕ_d are the values at the centroids of the upwind and downwind cells neighbouring the target face, drawn with a heavy line. The velocity vector $\mathbf{u}_f$ is prescribed at face f and determines the choice of stencil at each timestep.

such that $\mathrm{Co}_c \leq 1$ for all cells c in the domain. Hence, stability is constrained by the maximum Courant number of any cell in the domain.

The accurate approximation of the dependent variable at the face, ϕ_F, is key to the overall accuracy of the transport scheme. The cubicFit and linearUpwind schemes differ in their approximations of ϕ_F, and these approximation methods described next.

2.1. Cubic fit transport scheme

The cubicFit scheme approximates the value of the dependent variable at the face, ϕ_F, using a least squares fit over a stencil of surrounding cell centre values. To introduce the approximation method, we will consider how an approximate value is calculated for a face that is far away from the boundaries of a two-dimensional uniform rectangular mesh. For any mesh, every interior face connects two adjacent cells. The velocity direction at the face determines which of the two adjacent cells is the upwind cell. Since the stencil is upwind-biased, two stencils must be constructed for every interior face, and the appropriate stencil is chosen for each face depending on the velocity direction at that face for every timestep.

The upwind-biased stencil for a face f is shown in figure 1a. The wind at the face, $\mathbf{u}_f$, is blowing from the upwind cell c_u to the downwind cell c_d. To obtain an approximate value at f, a polynomial least squares fit is calculated using the stencil values. The stencil has 4 points in x and 3 points in y, leading to a natural choice of polynomial that is cubic in x and quadratic in y:

$$\phi = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 xy + a_6 y^2 + a_7 x^3 + a_8 x^2 y + a_9 x y^2 \tag{5}$$

A least squares approach is needed because the system of equations is overconstrained, with 12 stencil values but only 9 polynomial terms. The stencil geometry is expressed in a local coordinate system with the face centroid as the origin so that the approximated value ϕ_F is equal to the constant coefficient a_1.

The remainder of this subsection generalises the approximation technique for arbitrary meshes and describes the methods for constructing stencils, performing a least squares fit with a suitable polynomial, and ensuring numerical stability of the transport scheme.

2.1.1. Stencil construction

For every interior face, two stencils are constructed, one for each of the possible upwind cells. Stencils are not constructed for boundary faces because values of ϕ at boundaries are calculated from prescribed boundary conditions. For a given interior face f and upwind cell c_u, we find those faces that are connected to c_u and 'oppose' face f. These are called the *opposing faces*. The opposing faces for face f and upwind cell c_u are determined as follows. Defining G

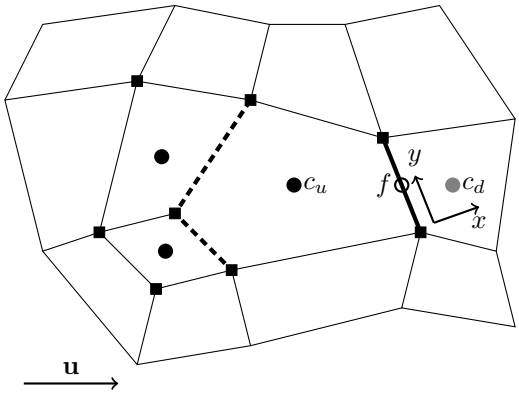

Figure 2: A thirteen-cell, upwind-biased stencil for face f connecting the pentagonal upwind cell, c_u, and the downwind cell c_d. The dashed lines denote the two faces of cell c_u that oppose f, and black circles mark the centroids of the internal cells that are connected to these two opposing faces. The stencil is extended outwards by including cells that share vertices with the three internal cells, where black squares mark these vertices. The local coordinate system (x, y) has its origin at the centroid of face f, marked by an open circle, with x normal to f and y perpendicular to x.

to be the set of faces other than f that border cell c_u, we calculate the 'opposedness', Opp, between faces f and $g \in G$, defined as

$$\mathrm{Opp}(f, g) \equiv -\frac{\mathbf{S}_f \cdot \mathbf{S}_g}{|\mathbf{S}_f|^2} \tag{6}$$

where $\mathbf{S}_f$ and $\mathbf{S}_g$ are the surface area vectors pointing outward from cell c_u for faces f and g respectively. Using the fact that $\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}|\,|\mathbf{b}| \cos(\theta)$ we can rewrite equation (6) as

$$\mathrm{Opp}(f, g) = -\frac{|\mathbf{S}_g|}{|\mathbf{S}_f|} \cos(\theta) \tag{7}$$

where θ is the angle between faces f and g. In this form, it can be seen that Opp is a measure of the relative area of g and how closely it parallels face f.

The set of opposing faces, OF, is a subset of G, comprising those faces with Opp ≥ 0.5, and the face with the maximum opposedness. Expressed in set notation, this is

$$\mathrm{OF}(f, c_u) \equiv \{g : \mathrm{Opp}(f, g) \geq 0.5\} \cup \{g : \max_{g \in G}(\mathrm{Opp}(f, g))\} \tag{8}$$

On a rectangular mesh, there is always one opposing face g that is exactly parallel to the face f such that $\mathrm{Opp}(f, g) = 1$.

Once the opposing faces have been determined, the set of internal and external cells must be found. The *internal cells* are those cells that are connected to the opposing faces. Note that c_u is always an internal cell. The *external cells* are those cells that share vertices with the internal cells. Note that c_d is always an external cell. Finally, the *stencil boundary faces* are boundary faces having Dirichlet boundary conditions[1] that share a vertex with the internal cells. Having found these three sets, the stencil is constructed to comprise all internal cells, external cells and stencil boundary faces.

Figure 2 illustrates a stencil construction for face f connecting upwind cell c_u and downwind cell c_d. The two opposing faces are denoted by thick dashed lines and the centres of the three adjoining internal cells are marked by black circles. The stencil is extended outwards by including the external cells that share vertices with the internal cells, where the vertices are marked by black squares. The resultant stencil contains 13 cells.

[1] Boundary faces with Neumann boundary conditions would require extrapolated boundary values to be calculated. This would create a feedback loop in which boundary values are extrapolated from interior values, then interior values are transported using stencils that include boundary values. We have not considered how such an extrapolation could made consistent with the multidimensional polynomial reconstruction. Using an inconsistent extrapolation, we found that the feedback loop is able to generate slow-growing numerical instabilities near some highly-distorted cells. Hence, boundary faces with Neumann boundary conditions are excluded from the set of stencil boundary faces.

2.1.2. Least squares fit

To approximate the value at a face f, a least squares fit is calculated from a stencil of surrounding cell centre values. First, we will show how a polynomial least squares fit is calculated for a face on a rectangular mesh. Second, we will make modifications to the least squares fit that are necessary for numerical stability. Finally, we will describe how the approach is applicable to faces of arbitrary meshes.

For faces that are far away from the boundaries of a rectangular mesh, we fit the multidimensional polynomial given by equation (5) that has nine unknown coefficients, $\mathbf{a} = a_1 \dots a_9$, using the twelve cell centre values from the upwind-biased stencil, $\boldsymbol{\phi} = \phi_1 \dots \phi_{12}$. This yields a matrix equation

$$\begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & x_1y_1 & y_1^2 & x_1^3 & x_1^2y_1 & x_1y_1^2 \\ 1 & x_2 & y_2 & x_2^2 & x_2y_2 & y_2^2 & x_2^3 & x_2^2y_2 & x_2y_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{12} & y_{12} & x_{12}^2 & x_{12}y_{12} & y_{12}^2 & x_{12}^3 & x_{12}^2y_{12} & x_{12}y_{12}^2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_9 \end{bmatrix} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{12} \end{bmatrix} \tag{9}$$

which can be written as

$$\mathbf{Ba} = \boldsymbol{\phi} \tag{10}$$

The rectangular matrix $\mathbf{B}$ has one row for each cell in the stencil and one column for each term in the polynomial. $\mathbf{B}$ is called the *stencil matrix*, and it is constructed using only the mesh geometry. A local coordinate system is established in which x is normal to the face f and y is perpendicular to x. The coordinates (x_i, y_i) give the position of the centroid of the ith cell in the stencil. A two-dimensional stencil is also used for the tests on spherical meshes in section 3.3. In these tests, cell centres are projected onto a plane that is tangent to the sphere at the face centre. The unknown coefficients $\mathbf{a}$ are calculated using the pseudo-inverse, $\mathbf{B}^+$, found by singular value decomposition:

$$\mathbf{a} = \mathbf{B}^+\boldsymbol{\phi} \tag{11}$$

Recall that the approximate value ϕ_F is equal to the constant coefficient a_1, which is a weighted mean of $\boldsymbol{\phi}$:

$$a_1 = \begin{bmatrix} b^+_{1,1} \\ b^+_{1,2} \\ \vdots \\ b^+_{1,12} \end{bmatrix} \cdot \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{12} \end{bmatrix} \tag{12}$$

where the weights $b^+_{1,1} \dots b^+_{1,12}$ are the elements of the first row of $\mathbf{B}^+$. Note that the majority of the least squares fit procedure depends on the mesh geometry only. An implementation may precompute the pseudo-inverse for each stencil during model initialisation, and only the first row needs to be stored. Knowledge of the values of $\boldsymbol{\phi}$ is only required to calculate the weighted mean given by equation (12), which is evaluated once per face per timestep.

In the least squares fit presented above, all stencil values contributed equally to the polynomial fit. It is necessary for numerical stability that the polynomial fits the cells connected to face f more closely than other cells in the stencil, as shown by [22, 23]. To achieve this, we allow each cell to make an unequal contribution to the least squares fit. We assign an integer *multiplier* to each cell in the stencil, $\mathbf{m} = m_1 \dots m_{12}$, and multiply equation (10) to obtain

$$\tilde{\mathbf{B}}\mathbf{a} = \mathbf{m} \cdot \boldsymbol{\phi} \tag{13}$$

where $\tilde{\mathbf{B}} = \mathbf{MB}$ and $\mathbf{M} = \mathrm{diag}(\mathbf{m})$. The constant coefficient a_1 is calculated from the pseudo-inverse, $\tilde{\mathbf{B}}^+$:

$$a_1 = \tilde{\mathbf{b}}^+_{\mathbf{1}} \cdot \mathbf{m} \cdot \phi \tag{14}$$

where $\tilde{\mathbf{b}}^+_{\mathbf{1}} = \tilde{b}^+_{1,1} \dots \tilde{b}^+_{1,12}$ are the elements of the first row of $\tilde{\mathbf{B}}^+$. Again, a_1 is a weighted mean of $\boldsymbol{\phi}$, where the weights are now $\tilde{\mathbf{b}}^+_{\mathbf{1}} \cdot \mathbf{m}$. Values for $\mathbf{m}$ are chosen so that the cells connected to face f make a greater contribution to the least squares fit, as discussed later in section 2.1.4.

For faces of a non-rectangular mesh, or faces that are near a boundary, the number of stencil cells and number of polynomial terms may differ: a stencil will have two or more cells and, for two-dimensional meshes, its polynomial will have between one and nine terms. Additionally, the polynomial cannot have more terms than its stencil has cells because this would lead to an underconstrained system of equations. The procedure for choosing suitable polynomials is discussed next.

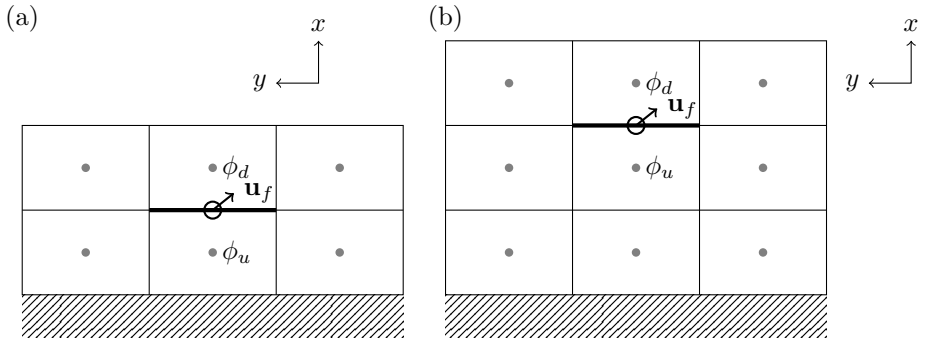

Figure 3: Upwind-biased stencils for faces near the lower boundary of a rectangular mesh, with (a) a 3×2 stencil for the face immediately adjacent to the lower boundary, and (b) a 3×3 stencil for the face immediately adjacent to the face in (a). For both stencils, attempting a least squares fit using the nine-term polynomial in equation (5) would result in an underconstrained problem.

2.1.3. Polynomial generation

The majority of faces on a uniform two-dimensional mesh have stencils with more than nine cells. For example, a rectangular mesh has 12 points (figure 1a), and a hexagonal mesh has 10 points (figure 1b). In both cases, constructing a system of equations using the nine-term polynomial in equation (5) leads to an overconstrained problem that can be solved using least squares. However, this is not true for faces near boundaries: stencils that have fewer than nine cells (figure 3a) would result in an underconstrained problem, and stencils that have exactly nine cells may lack sufficient information to constrain high-order terms. For example, the stencil in figure 3b lacks sufficient information to fit the x^3 term. In such cases, it becomes necessary to perform a least squares fit using a polynomial with fewer terms.

For every stencil, we find a set of *candidate polynomials* that do not result in an underconstrained problem. In two dimensions, a candidate polynomial has between one and nine terms and includes a combination of the terms in equation (5). There are two additional constraints that a candidate polynomial satisfy.

First, high-order terms may be included in a candidate polynomial only if the lower-order terms are also included. Let

$$M(x, y) = x^i y^j : i, j \geq 0 \text{ and } i + j \leq 3 \tag{15}$$

be the set of all monomials of degree at most 3 in x, y. A subset S of $M(x, y)$ is "dense" if, whenever $x^a y^b$ and $x^c y^d$ are in S with $a \leq c$ and $b \leq d$, then $x^i y^j$ is also in S for all $a < i < c$, $b < j < d$. For example, the polynomial $\phi = a_1 + a_2 x + a_3 y + a_4 xy + a_5 x^2 + a_6 x^2 y$ is a dense subset of $M(x, y)$, but $\phi = a_1 + a_2 x + a_3 y + a_4 x^2 y$ is not because $x^2 y$ can be included only if xy and x^2 are also included.

Second, a candidate polynomial must have a stencil matrix $\mathbf{B}$ that is full rank. The matrix is considered full rank if its smallest singular value is greater than 1×10^{-9}. Using a polynomial with all nine terms and the stencil in figure 3b results in a rank-deficient matrix and so the nine-term polynomial would not be a candidate polynomial.

The candidate polynomials are all the dense subsets of $M(x, y)$ that have a stencil matrix that is full rank. The final stage of the transport scheme selects a candidate polynomial and ensures that the least squares fit is numerically stable.

2.1.4. Stabilisation procedure

So far, we have constructed a stencil and found a set of candidate polynomials. Applying a least squares fit to any of these candidate polynomials avoids creating an underconstrained problem. The final stage of the transport scheme chooses a suitable candidate polynomial and appropriate multipliers $\mathbf{m}$ so that the fit is numerically stable.

The approximated value ϕ_F is equal to a_1 which is calculated from equation (14). The value of a_1 is a weighted mean of $\boldsymbol{\phi}$ where $\mathbf{w} = \tilde{\mathbf{b}}_1^+ \cdot \mathbf{m}$ are the weights. If the cell centre values $\boldsymbol{\phi}$ are assumed to approximate a smooth field then we expect ϕ_F to be close to the values of ϕ_u and ϕ_d, and expect ϕ_F to be insensitive to small changes in $\boldsymbol{\phi}$. When the weights $\mathbf{w}$ have large magnitude then this is no longer true: ϕ_F becomes sensitive to small changes in $\boldsymbol{\phi}$ which can result in large departures from the smooth field $\boldsymbol{\phi}$.

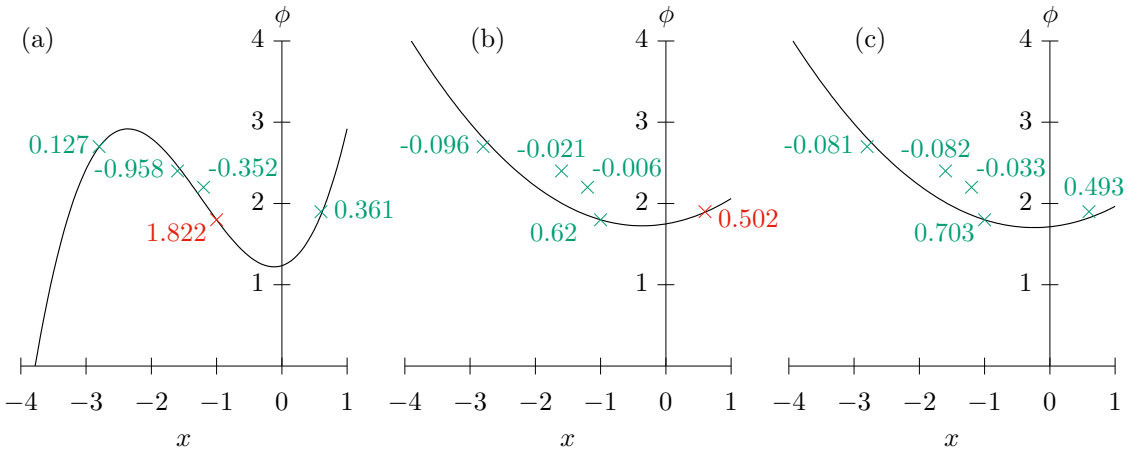

Figure 4: A one-dimensional least squares fits to a stencil of five points using (a) a cubic polynomial with multipliers $m_u = 1024$, $m_d = 1024$ and $m_p = 1$, (b) a quadratic polynomial with the same multipliers, and (c) a quadratic polynomial with multipliers $m_u = 1024$, $m_d = 1$ and $m_p = 1$. Notice that the curves in (a) and (b) fit almost exactly through the upwind and downwind points immediately adjacent to the y-axis, but in (c) the curve fits almost exactly only through the upwind point immediately to the left of the y-axis. The point data are labelled with their respective weights. Points that have failed one of the stability constraints in equation (16) are marked in red. The upwind point is located at $(-1, 1.8)$ and the downwind point at $(0.62, 1.9)$, and the peripheral points are at $(-2.8, 2.4)$, $(-1.6, 2.7)$ and $(-1.2, 2.2)$.

A one-dimensional von Neumann analysis was performed to obtain stability constraints on the weights $\mathbf{w}$. The analysis is presented in the appendix, and it shows that the weights must satisfy three constraints:

$$0.5 \leq w_u \leq 1 \tag{16a}$$

$$0 \leq w_d \leq 0.5 \tag{16b}$$

$$w_u - w_d \geq \max_{p \in P}(|w_p|) \tag{16c}$$

where w_u and w_d are the weights for the upwind and downwind cells respectively. The *peripheral cells* P are the cells in the stencil that are not the upwind or downwind cells, and w_p is the weight for a given peripheral cell p. The upwind, downwind and peripheral weights sum to one such that $w_u + w_d + \sum_{p \in P} w_p = 1$.

The stabilisation procedure comprises three steps. In the first step, the candidate polynomials are sorted in preference order so that candidates with the most terms are preferred over those with fewer terms. If there are multiple candidates with the same number of terms, the candidate with the largest minimum singular value of $\mathbf{B}$ is preferred. This ordering ensures that the preferred candidate is the highest-order polynomial with the most information content.

In the second step, the most-preferred polynomial is taken from the list of candidates and the multipliers are assigned so that the upwind cell and downwind cell have multipliers $m_u = 2^{10}$ and $m_d = 2^{10}$ respectively, and all peripheral cells have multipliers $m_p = 1$. These multipliers are very similar to those used by [22], leading to a well-conditioned matrix $\tilde{\mathbf{B}}$ and a least squares fit in which the polynomial passes almost exactly through the upwind and downwind cell centre values.

In the third step, we calculate the weights $\mathbf{w}$ and evaluate them against the stability constraints given in equation (16). If any constraint is violated, the value of m_d is halved and the constraints are evaluated with the new weights. This step is repeated until the weights satisfy the stability constraints, or m_d becomes smaller than one. In practice, the constraints are satisified when m_d is either small (between 1 and 4) or equal to 2^{10}. The upwind multiplier m_u is fixed at 2^{10}. If the constraints are still not satisfied, then we start again from the second step with the next-preferred polynomial in the candidate list.

Finally, if no stable weights are found for any candidate polynomial, we revert to an upwind scheme such that $w_u = 1$ and all other weights are zero. In fact, we have not encountered any stencil for which this last resort is required.

To illustrate the stabilisation procedure, figure 4a presents a one-dimensional example of a cubic polynomial fitted

through five points, with the weight at each point printed above it. In preference order, the candidate polynomials are

$$\phi = a_1 + a_2 x + a_3 x^2 + a_4 x^3 \tag{17}$$

$$\phi = a_1 + a_2 x + a_3 x^2 \tag{18}$$

$$\phi = a_1 + a_2 x \tag{19}$$

$$\phi = a_1 \tag{20}$$

We begin with the cubic equation. The multipliers are chosen so that the polynomial passes almost exactly through the upwind and downwind points that are immediately to the left and right of the y-axis respectively. The constraint on the upwind point is violated because $w_u = 1.822 > 1$ (equation 16a). Reducing the downwind multiplier does not help to satisfy the constraint, so we start again with the quadratic equation (figure 4b). Again, the multipliers are chosen to force the polynomial through the upwind and downwind points, but this violates the constraint on the downwind point because $w_d = 0.502 > 0.5$ (equation 16b). This time, however, stable weights are found by reducing w_d to one (figure 4c) and these are the weights that will be used to approximate ϕ_F, where the polynomial intercepts the y-axis.

2.2. Multidimensional linear upwind transport scheme

The multidimensional linear upwind scheme, called "linearUpwind" hereafter, is documented here since it provides a baseline accuracy for the experiments in section 3. The approximation of ϕ_F is calculated using a gradient reconstruction,

$$\phi_F = \phi_u + \nabla_c \phi \cdot \left(\mathbf{x}_f - \mathbf{x}_c\right) \tag{21}$$

where ϕ_u is the upwind value of ϕ, and $\mathbf{x}_f$ and $\mathbf{x}_c$ are the position vectors of the face centroid and cell centroid respectively. TODO: *the length of this vector should change when using the spherical correction, but doesn't. don't worry about this for now* The gradient $\nabla_c \phi$ is calculated using Gauss' theorem:

$$\nabla_c \phi = \frac{1}{\mathcal{V}_c} \sum_{f \in c} \tilde{\phi_F} \mathbf{S}_f \tag{22}$$

where $\tilde{\phi_F}$ is linearly interpolated from the two neighbouring cells of face f. For cells adjacent to boundaries having zero gradient boundary conditions, the boundary value is set to be equal to the cell centre value before equation (22) is evaluated. This implementation of the multidimensional linear upwind scheme is included in the OpenFOAM software distribution [34].

3. Results

3.1. Horizontal transport over mountains

TODO: *include convergence plot of horizontal advection over highly-distorted BTF grid, compare linearUpwind and cubicFit. Maybe use* $h_0 = 6\,\mathrm{km}$*. Should demonstrate that cubicFit is second-order even on highly-distorted meshes.*

3.2. Transport over a mountainous lower boundary

A two-dimensional transport test over mountains was developed in [2] to study the effect of terrain-following coordinate transformations on numerical accuracy. In this standard test, a tracer is positioned aloft and transported horizontally over wave-shaped terrain. This test presents no particular challenge on cut cell meshes because there is zero velocity and zero tracer density near the ground [35]. Here we present a variation of this standard test that challenges transport schemes on all mesh types. By positioning the tracer next to the ground and modifying the velocity field, we can assess the accuracy of the cubicFit scheme near the lower boundary. Results using the cubicFit scheme are compared with the linearUpwind scheme on basic terrain-following, cut cell and slanted cell meshes.

The domain is defined on a rectangular x–z plane that is 301 km wide and 25 km high as measured between parallel boundary edges. The domain is subdivided into a 301×50 mesh such that $\Delta x = 1\,\mathrm{km}$ and $\Delta z = 500\,\mathrm{m}$. A boundary condition of $\phi = 0\,\mathrm{kg\,m^{-3}}$ is imposed at the inlet boundary. The outlet, ground and top boundary conditions are

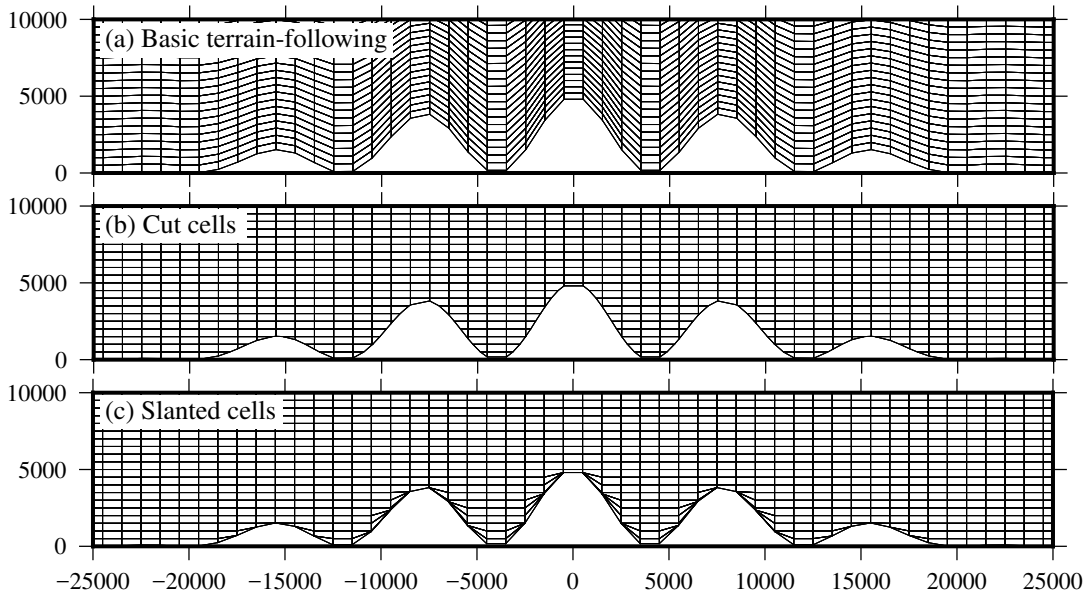

Figure 5: Cell edges of (a) basic terrain-following, (b) cut cell, and (c) slanted cell two-dimensional x–z meshes used for the tracer transport tests in section 3.2. The peak mountain height $h_0 = 5\,\text{km}$. Only the lowest 10 km for the central region of the domain in shown. The entire domain is 301 km wide and 25 km high.

$\partial\phi/\partial n = 0\,\text{kg}\,\text{m}^{-4}$ where the derivative is normal to the boundary. Since the cubicFit transport scheme uses values at boundaries with Dirichlet boundary conditions, only inlet boundary values are included in this test case.

The terrain is wave-shaped, specified by the surface height h such that

$$h(x) = h^\star \cos^2(\alpha x) \tag{23a}$$

where

$$h^\star(x) = \begin{cases} h_0 \cos^2(\beta x) & \text{if } |x| < a \\ 0 & \text{otherwise} \end{cases} \tag{23b}$$

where $a = 25\,\text{km}$ is the mountain envelope half-width, $h_0 = 5\,\text{km}$ is the maximum mountain height, $\lambda = 8\,\text{km}$ is the wavelength, $\alpha = \pi/\lambda$ and $\beta = \pi/(2a)$. Note that, in order to make this test more challenging, the mountain height h_0 is double the mountain height used by [2].

Basic terrain-following, cut cell and slanted cell meshes are constructed by modifying the uniform 301×50 mesh using this terrain profile. The basic terrain-following method uses a linear decay function so that mesh surfaces become horizontal at the top of the model domain [5],

$$z(x, y) = (H - h(x, y))\left(z^\star / H\right) + h(x, y) \tag{24}$$

where z is the geometric height at a horizontal point (x, y), H is the height of the domain, $h(x, y)$ is the surface elevation and $z^\star$ is the computational height of a mesh surface. If there was no terrain then $h = 0$ and $z = z^\star$. Cut cell meshes are constructed using the ASAM grid generator [36, 37]. Slanted cell meshes are constructed following the approach by [13]: vertices that are underground are moved up to the surface and zero-area faces and zero-volume cells are removed. Unlike [13], vertices are never moved downwards.

Cell edges in the central region of the domain are shown in figure 5 for each of the three mesh types. Cells in the BTF mesh are highly distorted over steep slopes (figure 5a) while the cut cell mesh (figure 5b) and slanted cell mesh (figure 5c) are orthogonal everywhere except for cells nearest the ground.

Similar to the approach by [13], a velocity field is chosen that is everywhere tangential to the surfaces of a terrain-following coordinate, and the coordinate is chosen so that flow is misaligned with the BTF, cut cell and slanted cell

	Peak mountain height h_0 (km)				
Mesh type	0	3	4	5	6
BTF	40	16	10	8	5
Cut cell	40	1.6	1.6	0.5	1.5
Slanted cell	40	8	6.25	5	4

Table 1: Time-steps (s) for the two-dimensional transport test over terrain. The time-steps were chosen so that the maximum Courant number was between 0.36 and 0.46.

meshes away from the ground. This ensures that flow always crosses mesh surfaces in order to challenge the transport scheme. This choice also ensures that there is no normal flow at the lower boundary. A streamfunction Ψ is used so that the discrete velocity field is non-divergent, such that

$$\Psi(x,z) = -u_0 H_1 \frac{z-h}{H_1-h} \tag{25}$$

where $u_0 = 10\,\mathrm{m\,s^{-1}}$, which is the horizontal velocity where $h(x) = 0$. The velocity field becomes horizontal at $H_1 = 10\,\mathrm{km}$. Note that H_1 is chosen to be much smaller than the domain height H in equation (24) so that flow crosses the surfaces of the BTF mesh. The horizontal and vertical components of velocity, u and w, are then given by

$$u = -\frac{\partial \Psi}{\partial z} = u_0 \frac{H_1}{H_1-h}, \quad w = \frac{\partial \Psi}{\partial x} = u_0 H_1 \frac{\mathrm{d}h}{\mathrm{d}x}\frac{H_1-z}{(H_1-h)^2} \tag{26}$$

$$\frac{\mathrm{d}h}{\mathrm{d}x} = -h_0\left[\beta\cos^2(\alpha x)\sin(2\beta x) + \alpha\cos^2(\beta x)\sin(2\alpha x)\right] \tag{27}$$

The flow is predominantly horizontal with non-zero vertical velocities only above sloping terrain. Unlike the horizontal transport test in [2], the velocity field presented here extends from the top of the domain all the way to the ground.

At $t = 0\,\mathrm{s}$, a tracer with density ϕ is positioned upwind of the mountain at the ground. It has the shape

$$\phi(x,z) = \phi_0 \begin{cases} \cos^2\left(\frac{\pi r}{2}\right) & \text{if } r \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{28}$$

with radius r given by

$$r = \sqrt{\left(\frac{x-x_0}{A_x}\right)^2 + \left(\frac{z-z_0}{A_z}\right)^2} \tag{29}$$

where $A_x = 25\,\mathrm{km}$, $A_z = 10\,\mathrm{km}$ are the horizontal and vertical half-widths respectively, and $\phi_0 = 1\,\mathrm{kg\,m^{-3}}$ is the maximum density of the tracer. At $t = 0\,\mathrm{s}$, the tracer is centred at $(x_0, z_0) = (-50\,\mathrm{km}, 0\,\mathrm{km})$ so that the tracer is upwind of the mountain and centred at the ground.

Tests are integrated forward for 10 000 s, by which time the tracer has moved downwind of the mountain. Different time-steps were chosen for each mesh so that the maximum Courant number was about 0.4 (table 1). An analytic solution at 10 000 s is obtained by calculating the new horizontal position of the tracer. Integrating along the trajectory yields t, the time taken to move from the left side of the mountain to the right:

$$\mathrm{d}t = \mathrm{d}x/u(x) \tag{30}$$

$$t = \int_0^x \frac{H-h(x)}{u_0 H}\,\mathrm{d}x \tag{31}$$

$$t = \frac{x}{u_0} - \frac{h_0}{16 u_0 H}\left[4x + \frac{\sin 2(\alpha+\beta)x}{\alpha+\beta} + \frac{\sin 2(\alpha-\beta)x}{\alpha-\beta} + 2\left(\frac{\sin 2\alpha x}{\alpha} + \frac{\sin 2\beta x}{\beta}\right)\right] \tag{32}$$

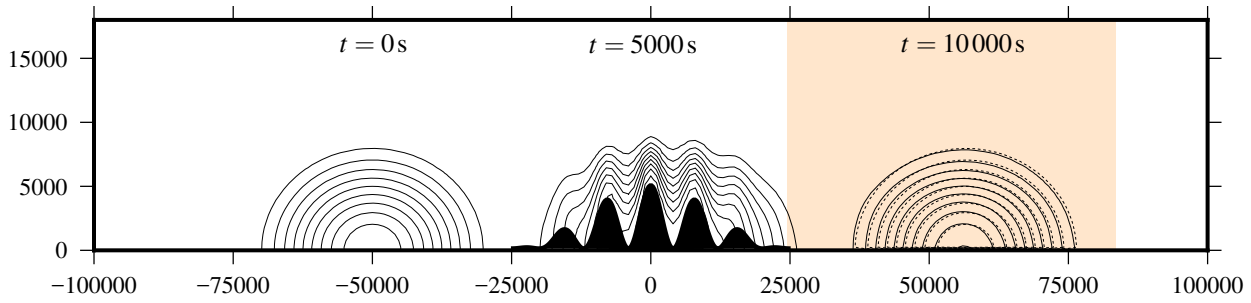

Figure 6: Evolution of the tracer in the two-dimensional transport test over steep terrain. The tracer is transported to the right over the wave-shaped terrain. Tracer contours are every $0.1\,\mathrm{kg\,m^{-3}}$. The result obtained using the cubicFit scheme on the basic terrain-following mesh is shown at $t = 0\,\mathrm{s}$, $t = 5000\,\mathrm{s}$ and $t = 10\,000\,\mathrm{s}$ with solid black contours. The analytic solution at $t = 10\,000\,\mathrm{s}$ is shown with dotted contours. The shaded box indicates the region that is plotted in figure 7.

By solving this equation we find that $x(t = 10\,000\,\mathrm{s}) = 54\,342.8\,\mathrm{m}$.

Tracer contours at the initial time $t = 0\,\mathrm{s}$, half-way time $t = 5000\,\mathrm{s}$, and end time $t = 10\,000\,\mathrm{s}$ are shown in figure 6 using the cubicFit scheme on the BTF mesh. As apparent at $t = 5000\,\mathrm{s}$, the tracer is distorted by the terrain-following velocity field as it passes over the mountain, but its original shape is restored once it has cleared the mountain by $t = 10\,000\,\mathrm{s}$. A small phase lag is apparent when the numerical solution marked with solid contour lines is compared with the analytic solution marked with dotted contour lines.

Numerical errors are more clearly revealed by subtracting the analytic solution from the numerical solution. This enables the calculation of error norms,

$$\ell_2 = \sqrt{\frac{\sum_c (\phi - \phi_T)^2 \mathcal{V}_c}{\sum_c \left(\phi_T^2 \mathcal{V}_c\right)}} \tag{33}$$

$$\ell_\infty = \frac{\max_c |\phi - \phi_T|}{\max_c |\phi_T|} \tag{34}$$

where ϕ is the numerical value, ϕ_T is the analytic value, $\sum_c$ denotes a summation over all cells c in the domain, and $\max_c$ denotes a maximum value of any cell.

Errors are compared between BTF, cut cell and slanted cell meshes using the linearUpwind scheme (figures 7a, 7b and 7c respectively) and the cubicFit scheme (figures 7d, 7e and 7f respectively). Results are least accurate using the linearUpwind scheme on the slanted cell mesh (figure 7c). The final tracer is slightly distorted and does not extend far enough towards the ground. The error magnitude is reduced by using the linearUpwind scheme on the cut cell mesh (figure 7b), but the error's shape remains the same. The cubicFit scheme is less sensitive to the choice of mesh with similar error magnitudes on the BTF mesh (figure 7d), cut cell mesh (figure 7e) and slanted cell mesh (figure 7f). Errors using the cubicFit scheme on cut cell and slanted cell meshes are much smaller than the errors using the linearUpwind scheme on the same meshes.

To further examine the performance of the cubicFit scheme in the presence of steep terrain, a second series of tests were performed in which the peak mountain height was varied from 0 km to 6 km keeping all other parameters constant. Results were obtained on BTF, cut cell and slanted cell meshes using the linearUpwind scheme and cubicFit scheme. Again, the time-step was chosen for each test so that the maximum Courant number was about 0.4 (table 1). The ℓ_2 error for these tests is presented in figure 8. In all cases, error increases with increasing mountain height because steeper slopes lead to greater mesh distortions. Errors are identical for a given transport scheme when $h_0 = 0\,\mathrm{km}$ and the ground is entirely flat because the BTF, cut cell and slanted cell meshes are identical. Compared with the cubicFit scheme, the linearUpwind scheme is more sensitive to the mesh type and mountain height. The linearUpwind scheme is unstable on the slanted cell mesh with a peak mountain height $h_0 = 6\,\mathrm{km}$ despite using a stable Courant number of 0.428. In contrast, the cubicFit scheme is less sensitive to the mesh type and errors grow more slowly with increasing mountain height. The cubicFit scheme yields stable results in all tests.

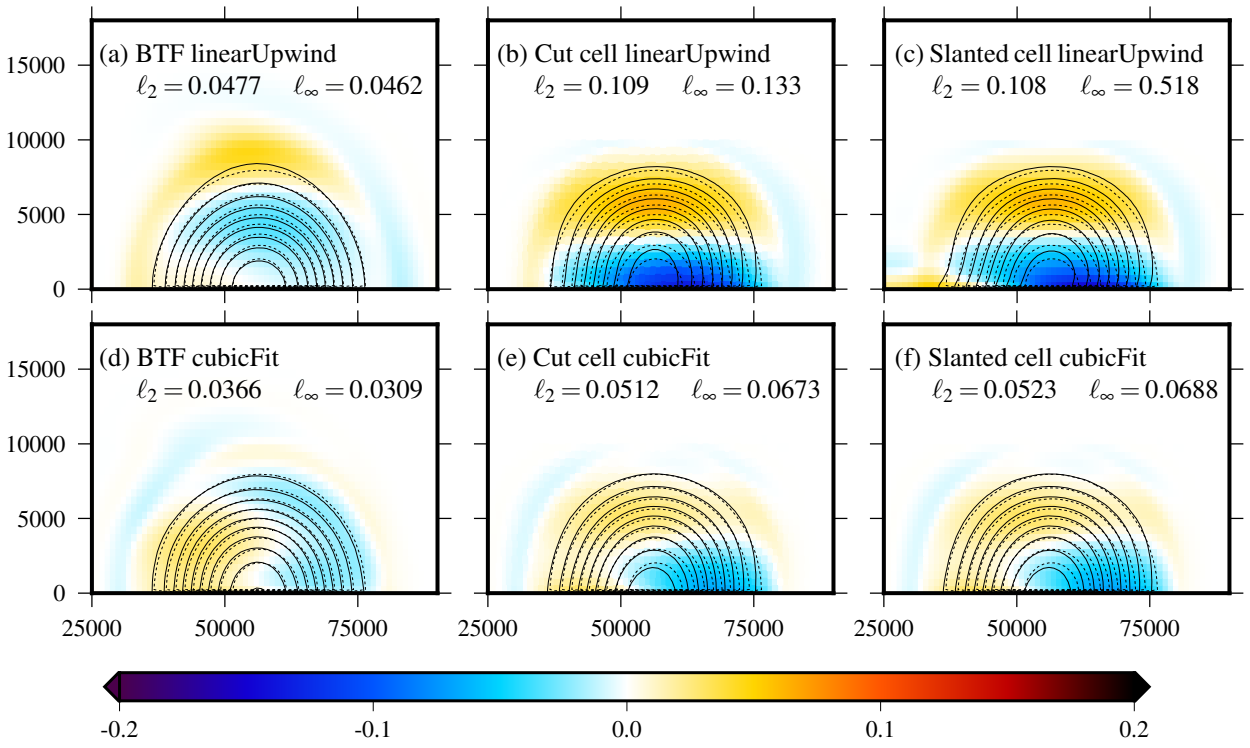

Figure 7: Tracer contours at t = 10 000 s for the two-dimensional tracer transport tests. A region in the lee of the mountain is plotted corresponding to the shaded area in figure 6. Results are presented on BTF, cut cell and slanted cell meshes (shown in figure 5) using the linearUpwind and cubicFit transport schemes. The numerical solutions are marked by solid black lines. The analytic solution is marked by dotted lines. Contours are every 0.1 kg m^{-3}.

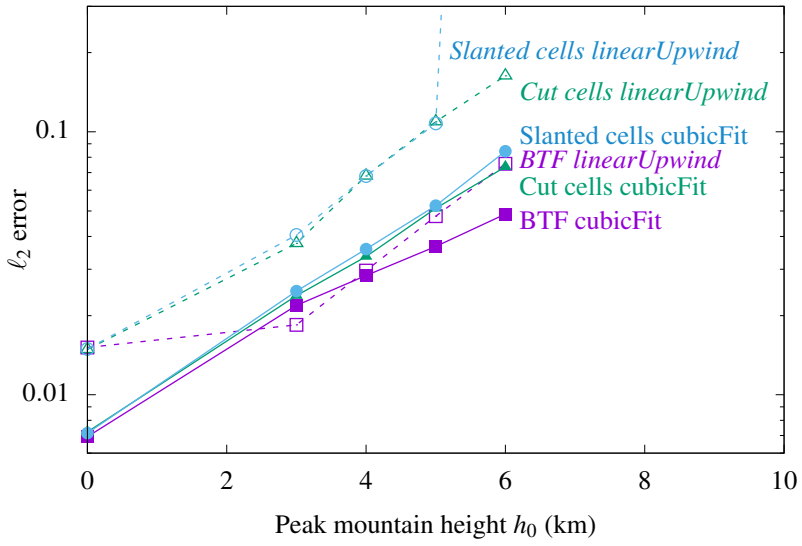

Figure 8: Error measures for the two-dimensional tracer transport tests with peak mountain heights h_0 from 0 km to 6 km. Results are compared on BTF, cut cell and slanted cell meshes using the linearUpwind and cubicFit schemes. At h_0 = 0 km the terrain is entirely flat and the BTF, cut cell and slanted cell meshes are identical. At h_0 = 6 km the linearUpwind scheme is unstable on the slanted cell mesh.

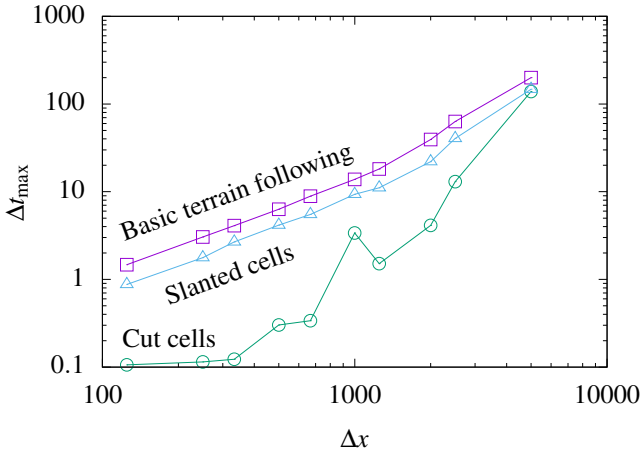

Figure 9: Longest stable timesteps, $\Delta t_{\max}$, for the two-dimensional tracer transport test on basic terrain-following, cut cell and slanted cell meshes at mesh spacings between $\Delta x = 5000$ m and $\Delta x = 125$ m. The tests were integrated with a maximum Courant number close to 1, while $\Delta t_{\max}$ is calculated as the timestep corresponding to a maximum Courant number of exactly 1.

A final series of tests were performed on BTF, slanted cell and cut cell meshes using the cubicFit scheme with a variety of mesh spacings between $\Delta x = 5000$ m and $\Delta x = 125$ m. Δz was chosen so that a constant aspect ratio is preserved such that $\Delta x/\Delta z = 2$. In order to verify that cubicFit is accurate near the stability limit, timesteps were chosen so that the maximum Courant number was close to one. Stable results were obtained in all tests and the cubic scheme was largely insensitive to the choice of timestep.

This series of tests also enables a comparison of longest stable timesteps between mesh types. The longest stable timestep for a maximum Courant number of one can be calculated as $\Delta t_{\max} = \Delta t / \max(\text{Co})$ where Δt is the timestep used in a particular test run and max(Co) is the maximum Courant number for that test run. The longest stable timesteps for BTF, cut cell and slanted cell meshes are presented in figure 9. BTF meshes permit the longest timesteps of all three meshes since cells are almost uniform in volume. As expected, the longest stable timestep scales linearly with BTF mesh spacing. There is no such linear scaling on cut cell meshes because these meshes can have arbitrarily small cells. The timestep constraints on cut cell meshes are the most severe of the three mesh types. Slanted cell meshes have a slightly stronger timestep constraint than BTF meshes, but still exhibit the same, predictable linear scaling with mesh spacing.

The transport tests presented in this section demonstrate that the cubicFit scheme is suitable for flows over very steep terrain on two-dimensional terrain-following, cut cell and slanted cell meshes. The cubicFit scheme is less sensitive to the mesh type and mountain steepness compared to the linearUpwind scheme. The linearUpwind scheme becomes unstable over very steep slopes but the cubicFit scheme is stable for all tests. In the next section, we evaluate the cubicFit scheme using more complex, deformational flows on icosahedral meshes and cubed-sphere meshes.

3.3. Deformational flow on a sphere

TODO: *check timestep values and Courant numbers and state these.*

To ensure that the cubicFit transport scheme is suitable for complex flows on a variety of meshes, we use a standard test of deformational flow on a spherical Earth [33]. Results are compared between linearUpwind and cubicFit schemes using hexagonal icosahedra and cubed-spheres. Hexagonal-icosahedral meshes are constructed by successive refinement of a regular icosahedron following the approach by [25]. Figure 10a shows an example of such a mesh that has been refined three times. Cubed-sphere meshes are constructed using an equi-distant gnomic projection of a cube having a uniform Cartesian mesh on each panel [38]. Figure 10b shows an example of a cubed-sphere mesh having panels with 8×8 cells.

Following appendix A9 in [28], the average equatorial spacing $\Delta\lambda$ is used as a measure of mesh spacing. It is

(a) Hexagonal icosahedral mesh (b) Cubed-sphere mesh

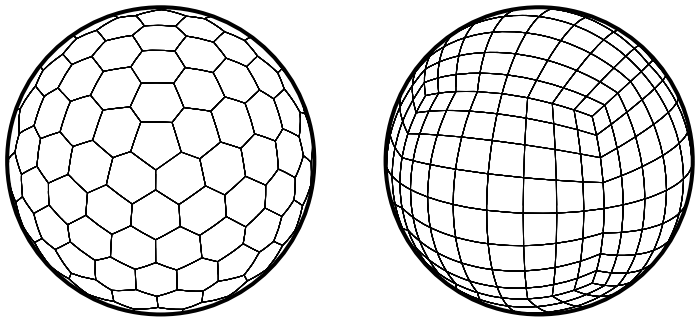

Figure 10: Illustrations of coarse meshes of a spherical Earth using (a) a hexagonal icosahedron that has been refined three times, and (b) a cubed-sphere with each panel having 8×8 cells.

defined as

$$\Delta\lambda = 360^\circ \frac{\overline{\Delta x}}{2\pi R_e} \tag{35}$$

where $\overline{\Delta x}$ is the mean distance between cell centres and $R_e = 6.3712 \times 10^6$ m is the radius of the Earth.

The deformational flow test in [33] comprised six elements:

1. a convergence test using a Gaussian-shaped tracer
2. a "minimal" resolution test using a cosine-shaped tracer
3. a test of filament preservation
4. a test using a "rough" slotted cylinder tracer
5. a test of correlation preservation between two tracers
6. a test using a divergent velocity field

We assess the cubicFit scheme using the first two tests only. We do not consider filament preservation, correlation preservation, or the transport of a "rough" slotted cylinder because no shape-preserving filter has yet been developed for cubicFit. Stable results were obtained when testing cubicFit using a divergent velocity field, but no further analysis is made here. TODO: *provide results as supplementary material?*

The first deformational flow test uses a C^∞ initial tracer that is transported in a non-divergent, time-varying rotational velocity field. The velocity field deforms two Gaussian 'hills' of tracer into thin vortical filaments. Half-way through the integration the rotation reverses so that the filaments become circular hills once again. The analytic solution at the end of integration is identical to the initial condition. A rotational flow is superimposed on a time-invariant background flow in order to avoid error cancellation. The non-divergent velocity field is defined by the streamfunction Ψ:

$$\Psi(\lambda, \theta, t) = \frac{10R_e}{T} \sin^2(\lambda') \cos^2(\theta) \cos\left(\frac{\pi t}{T}\right) - \frac{2\pi R_e}{T} \sin(\theta) \tag{36}$$

where λ is a longitude, θ is a latitude, $T = 1.0368 \times 10^6$ s is the duration of integration, and $\lambda' = \lambda - 2\pi t/T$.

The initial tracer ϕ is defined as the sum of two Gaussian hills:

$$\phi = \phi_1(\lambda, \theta) + \phi_2(\lambda, \theta) \tag{37}$$

An individual hill ϕ_i is given by

$$\phi_i(\lambda, \theta) = \phi_0 \exp\left(-b\left(\frac{|\mathbf{x} - \mathbf{x}_i|}{R_e}\right)^2\right) \tag{38}$$

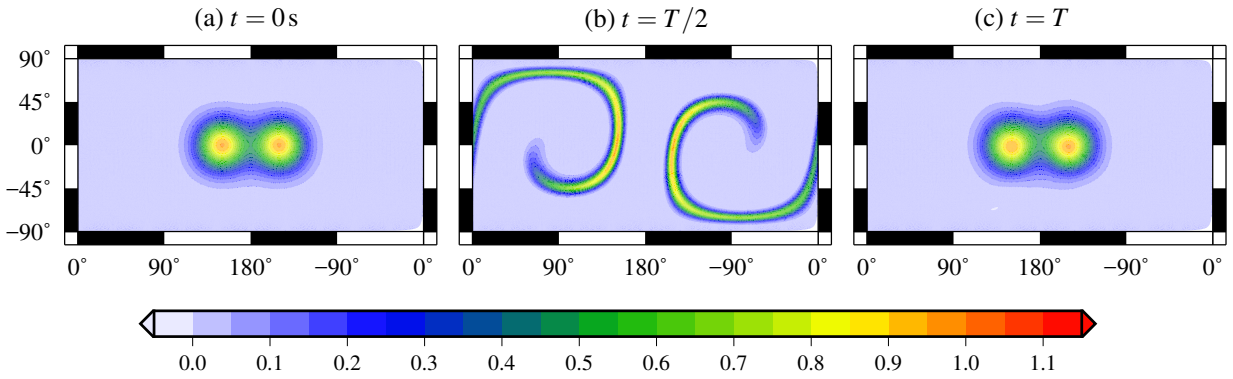

Figure 11: Tracer fields for the deformational flow test using initial Gaussian hills. The tracer is deformed by the velocity field before the rotation reverses to return the tracer to its original distribution: (a) the initial tracer distribution at $t = 0\,\mathrm{s}$; (b) by $t = T/2$ the Gaussian hills are stretched into a thin S-shaped filament; (c) at $t = T$ the tracer resembles the initial Gaussian hills except for some distortion and diffusion due to numerical errors. TODO: *replot using hex-9*

where $\phi_0 = 0.95$ and $b = 5$. The Cartesian position vector $\mathbf{x} = (x, y, z)$ is related to the spherical coordinates (λ, θ) by

$$(x, y, z) = (R_e \cos\theta \cos\lambda, R_e \cos\theta \sin\lambda, R_e \sin\theta) \tag{39}$$

The centre of hill i is positioned at $\mathbf{x}_i$. In spherical coordinates, two hills are centred at

$$(\lambda_1, \theta_1) = (5\pi/6, 0) \tag{40}$$

$$(\lambda_2, \theta_2) = (7\pi/6, 0) \tag{41}$$

The results in figure 11 are obtained using the cubicFit scheme on a hexagonal-icosahedral mesh with $\Delta\lambda = 0.271°$. The initial Gaussian hills are shown in figure 11a. At $t = T/2$ the tracer has been deformed into an S-shaped filament (figure 11b). By $t = T$ the tracer has almost returned to its original distribution except for some slight distortion and diffusion that are the result of numerical errors (figure 11c).

To determine the order of convergence and relative accuracy of the linearUpwind and cubicFit schemes, the same test was performed at a variety of mesh spacings betweeen $\Delta\lambda = 8.61°$ and $\Delta\lambda = 0.271°$ on hexagonal icosahedra and cubed-sphere meshes. The results are shown in figure 12. The solution is slow to converge at coarse resolutions, but both linearUpwind and cubicFit schemes achieve second-order accuracy at smaller mesh spacings. This behaviour agrees with the results from Lauritzen et al. [33]. For any given mesh type and mesh spacing, cubicFit is more accurate than linearUpwind. The linearUpwind scheme achieves more accurate results on hexagonal icosahedra compared to cubed-sphere meshes, but cubicFit is less sensitive to the mesh type.

A slightly more challenging variation of the same test is performed using a quasi-smooth tracer field is defined as the sum of two cosine bells,

$$\phi = \begin{cases} b + c\phi_1(\lambda, \theta) & \text{if } r_1 < r \\ b + c\phi_2(\lambda, \theta) & \text{if } r_2 < r \\ b & \text{otherwise} \end{cases} \tag{42}$$

The velocity field is the same as before. This test is used to determine the "minimal" resolution, which is specified by Lauritzen et al. [33] as the coarsest mesh spacing for which $\ell_2 \approx 0.033$. TODO: *minimal resolution for cubicFit on hex mesh is about $\Delta\lambda = 0.3°$* TODO: *and linearUpwind? and perhaps compare to similar schemes. mention that [28] has collated minimal resolutions for a number of other schemes.*

4. Conclusion

We have presented a new multidimensional method-of-lines transport scheme, cubicFit, that is suitable for complex flows on a wide variety of mesh types. Constraints derived from a von Neumann stability analysis are applied during

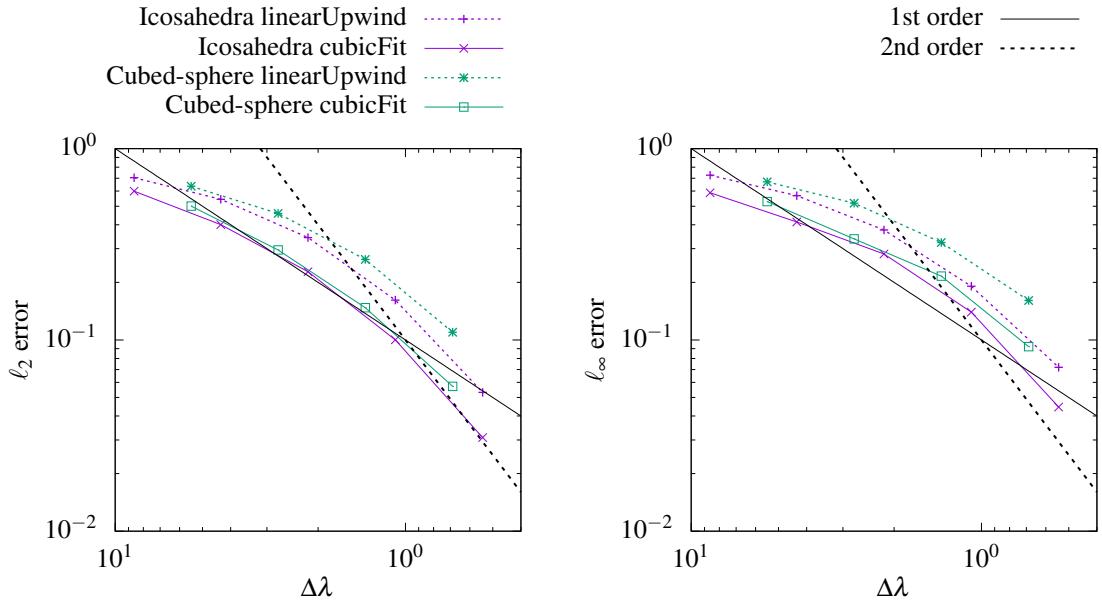

Figure 12: Numerical convergence of the deformational flow test on the sphere using initial Gaussian hills. ℓ_2 errors (equation 33) and ℓ_∞ errors (equation 34) are marked at mesh spacings between 8.61° and 0.271° using linearUpwind and cubicFit transport schemes on hexagonal icosahedra and cubed-sphere meshes.

model initialisation to make the scheme stable over steep terrain on arbitrarily-structured meshes. The scheme has a low computational cost at runtime, requiring only n multiplies per face per time-step using a stencil with n cells, with more expensive computations depending on the mesh geometry only.

The cubicFit scheme was compared to a multidimensional linear upwind scheme using two idealised numerical tests. The first test transported a tracer over steep slopes on a two-dimensional x–z plane using terrain-following, cut cell and slanted cell meshes. The cubicFit scheme was generally insensitive to the type of mesh and less sensitive to terrain steepness compared to linearUpwind. The cubicFit scheme maintained accuracy up to the stability limit of a Courant number of one. The second test evaluated the transport schemes in a standard deformational flow field on hexagonal icosahedra and cubed-sphere meshes. In all tests, compared to the linearUpwind scheme, the cubicFit scheme was more stable and more accurate.

TODO: *I haven't said anything about convergence*

5. Acknowledgements

James Shaw acknowledges support from a PhD studentship funded jointly by NERC grant NE/K500860/1 and the University of Reading with CASE support from the Met Office. We are grateful to the Leibniz Institute for Tropospheric Research for providing their cut cell mesh generator, and to Dr Shing Hing Man for his assistance with candidate polynomial generation. We also thank Dr Tristan Pryer for useful discussions about the cubicFit transport scheme.

Appendix A: One-dimensional von Neumann stability analysis

Two analyses are performed in order to find stability constraints on the weights $\mathbf{w} = \tilde{\mathbf{b}}_1^+ \cdot \mathbf{m}$ as appear in equation (14). The first analysis uses two points to derive separate constraints on the upwind weight w_u and downwind weight w_d. The second analysis uses three points to derive a constraint that considers all weights in a stencil.

Two-point analysis

We start with the conservation equation for a dependent variable ϕ that is discrete-in-space and continuous-in-time

$$\frac{\partial \phi_j}{\partial t} = -u\frac{\phi_R - \phi_L}{\Delta x} \tag{43}$$

where the left and right fluxes, ϕ_L and ϕ_R are weighted averages of the neighbouring points. Assuming that u is positive

$$\phi_L = \alpha_u \phi_{j-1} + \alpha_d \phi_j \tag{44}$$

$$\phi_R = \beta_u \phi_j + \beta_d \phi_{j+1} \tag{45}$$

where α_u and β_u are the upwind weights and α_d and β_d are the downwind weights for the left and right fluxes respectively, and $\alpha_u + \alpha_d = 1$ and $\beta_u + \beta_d = 1$. A subscript j denotes the value at a given point $x = j\Delta x$ where Δx is a uniform mesh spacing.

At a given time $t = n\Delta t$ at time-level n and with a time-step Δt, we assume a wave-like solution with an amplification factor A, such that

$$\phi_j^{(n)} = A^n e^{ijk\Delta x} \tag{46}$$

where $\phi_j^{(n)}$ denotes a value of ϕ at position j and time-level n. Using this to rewrite the left-hand side of equation (43)

$$\frac{\partial \phi_j}{\partial t} = \frac{\partial}{\partial t}\left(A^{t/\Delta t}\right) e^{ijk\Delta x} = \frac{\ln A}{\Delta t} A^n e^{ikj\Delta x} \tag{47}$$

hence equation (43) becomes

$$\frac{\ln A}{\Delta t} = -\frac{u}{\Delta x}\left(\beta_u + \beta_d e^{ik\Delta x} - \alpha_u e^{-ik\Delta x} - \alpha_d\right) \tag{48}$$

$$\ln A = -c\left(\beta_u - \alpha_d + \beta_d \cos k\Delta x + i\beta_d \sin k\Delta x - \alpha_u \cos k\Delta x + i\alpha_u \sin k\Delta x\right) \tag{49}$$

where the Courant number $c = u\Delta t/\Delta x$. Let $\mathfrak{R} = \beta_u - \alpha_d + \beta_d \cos k\Delta x - \alpha_u \cos k\Delta x$ and $\mathfrak{I} = \beta_d \sin k\Delta x + \alpha_u \sin k\Delta x$, then

$$\ln A = -c\left(\mathfrak{R} + i\mathfrak{I}\right) \tag{50}$$

$$A = e^{-c\mathfrak{R}} e^{-ic\mathfrak{I}} \tag{51}$$

and the complex modulus and complex argument of A are found to be

$$|A| = e^{-c\mathfrak{R}} = \exp\left(-c\left(\beta_u - \alpha_d + (\beta_d - \alpha_u)\cos k\Delta x\right)\right) \quad \text{and} \tag{52}$$

$$\arg(A) = -c\mathfrak{I} = -c\left(\beta_d + \alpha_u\right)\sin k\Delta x \tag{53}$$

For stability, we need $|A| \leq 1$ and for advection in the correct direction we need $\arg(A) < 0$ for $c > 0$, so

$$\beta_u - \alpha_d + (\beta_d - \alpha_u)\cos k\Delta x \geq 0 \quad \forall k\Delta x \quad \text{and} \tag{54}$$

$$\beta_d + \alpha_u > 0 \tag{55}$$

Imposing the additional constraints that $\alpha_u = \beta_u$ and $\alpha_d = \beta_d$:

$$|A| = \exp\left(-c\left(\alpha_u - \alpha_d\right)\left(1 - \cos k\Delta x\right)\right) \tag{56}$$

and given $1 - \cos k\Delta x \geq 0$, then

$$\alpha_u - \alpha_d \geq 0 \tag{57}$$

which provides a lower bound on α_u:

$$\alpha_u \geq \alpha_d \tag{58}$$

Additionally, we do not want more damping than an upwind scheme (where $\alpha_u = \beta_u = 1$, $\alpha_d = \beta_d = 0$), having an amplification factor, A_{up}:

$$|A_{\text{up}}| = \exp\left(-c\left(1 - \cos k\Delta x\right)\right) \tag{59}$$

So we need $|A| \geq |A_{\text{up}}|$:

$$-c\left(\alpha_u - \alpha_d\right)\left(1 - \cos k\Delta x\right) \geq -c\left(1 - \cos k\Delta x\right) \tag{60}$$

$$\alpha_u - \alpha_d \leq 1 \tag{61}$$

$$\alpha_u \leq 1 + \alpha_d \tag{62}$$

which provides an upper bound on α_u. Combining with eqn (58) we can bound α_u on both sides:

$$\alpha_d \leq \alpha_u \leq 1 + \alpha_d \tag{63}$$

Now, knowing that $\alpha_u + \alpha_d = 1$ (or $\alpha_d = 1 - \alpha_u$), then

$$1 - \alpha_u < \alpha_u \leq 1 + (1 - \alpha_u) \tag{64}$$

$$0.5 \leq \alpha_u \leq 1 \tag{65}$$

and, since $\alpha_u + \alpha_d = 1$, then

$$0 \leq \alpha_d \leq 0.5 \tag{66}$$

Three-point analysis

We start again from equation (43) but this time approximate ϕ_L and ϕ_R using three points:

$$\phi_L = \alpha_{uu}\phi_{j-2} + \alpha_u\phi_{j-1} + \alpha_d\phi_j \tag{67}$$

$$\phi_R = \alpha_{uu}\phi_{j-1} + \alpha_u\phi_j + \alpha_d\phi_{j+1} \tag{68}$$

having used the same weights α_{uu}, α_u and α_d for both left and right fluxes. Substituting equation (46) into equation (43) we find

$$A = \exp\left(-c\left[\alpha_{uu}\left(e^{-ik\Delta x} - e^{-2ik\Delta x}\right) + \alpha_u\left(1 - e^{-ik\Delta x}\right) + \alpha_d\left(e^{ik\Delta x} - 1\right)\right]\right) \tag{69}$$

So that, if the complex modulus $|A| \leq 1$ then

$$\alpha_u - \alpha_d + (\alpha_{uu} - \alpha_u + \alpha_d)\cos k\Delta x - \alpha_{uu}\cos 2k\Delta x \geq 0 \tag{70}$$

If $\cos k\Delta x = -1$ and $\cos 2k\Delta x = 1$ then $\alpha_u - \alpha_d \geq \alpha_{uu}$, and if $\cos k\Delta x = 0$ and $\cos 2k\Delta x = -1$ then $\alpha_u - \alpha_d \geq -\alpha_{uu}$. Hence we find that

$$\alpha_u - \alpha_d \geq |\alpha_{uu}| \tag{71}$$

and, when the same analysis is performed with four points, α_{uuu}, α_{uu}, α_u and α_d, we find that the same condition holds replacing α_{uu} with α_{uuu}. Hence, we generalise equation (71) to find the final stability constraint

$$\alpha_u - \alpha_d \geq \max_{p \in P} |\alpha_p| \tag{72}$$

where the peripheral cells P is the set of all stencil cells except for the upwind and downwind cell, and α_p is the weight for a given peripheral cell p.

Appendix B: Mesh geometry on a spherical Earth

The cubicFit transport scheme is implemented using the OpenFOAM CFD library. Unlike many atmospheric models that use spherical coordinates, OpenFOAM uses global, three-dimensional Cartesian coordinates. In order to perform the experiments on a spherical Earth presented in section 3.3, it is necessary for velocity fields and mesh geometries to be expressed in these global Cartesian coordinates.

Velocity field specification

The non-divergent velocity field in section 3.3 is specified as a streamfunction $\Psi(\lambda, \theta)$. Instead of calculating velocity vectors, the flux $\mathbf{u}_f \cdot \mathbf{S}_f$ through a face f is calculated directly from the streamfunction,

$$\mathbf{u}_f \cdot \mathbf{S}_f = \sum_{e \in f} \mathbf{e} \cdot \mathbf{x}_e \Psi(e) \tag{73}$$

where $e \in f$ denotes the edges e of face f, $\mathbf{e}$ is the edge vector joining its two vertices, $\mathbf{x}_e$ is the position vector of the edge midpoint, and $\Psi(e)$ is the streamfunction evaluated at the same position. Edge vectors are directed in a counter-clockwise orientation.

Spherical mesh construction

Since OpenFOAM does not support two-dimensional spherical meshes, instead, we construct meshes that have a single layer of cells that are 2000 m deep, having an inner radius $r_1 = R_e - 1000\,\text{m}$ and an outer radius $r_2 = R_e + 1000\,\text{m}$. By default, OpenFOAM meshes comprise polyhedral cells with straight edges and flat faces. This is problematic for spherical meshes because face areas and cell volumes are too small. For tests on a spherical Earth, we override the default configuration and calculate our own face areas, cell volumes, face centres and cell centres that account for the spherical geometry.

A face is assumed to be either a surface face or radial face. A surface face has any number of vertices, all of equal radius. A radial faces have four vertices with two different radii, r_1 and r_2, and two different horizontal coordinates, (λ_1, θ_1) and (λ_2, θ_2). A radial face centre is modified so that it has a radius R_e. The latitudinal and longitudinal components of a radial face centre need no modification. TODO: *although the code does recalculate this anyway!* The face area A_f for a radial face f is the area of the annular sector,

$$A_f = \frac{d}{2} \left| r_2^2 - r_1^2 \right| \tag{74}$$

where d is the great-circle distance between (λ_1, θ_1) and (λ_2, θ_2).

To calculate the centre of a surface face f, a new vertex is created that is positioned at the mean of the face vertices. Note that this centre position, $\tilde{\mathbf{c}}_f$, is used in intermediate calculations and it is not the face centre position. Next, the surface face is subdivided into spherical triangles that share this new vertex. The face centre direction and radius are calculated separately. The face centre direction $\hat{\mathbf{r}}$ is the mean of the spherical triangle centres weighted by their solid angle

$$\hat{\mathbf{r}} = \frac{\sum_{t \in f} \Omega_t \left(\mathbf{x}_{t,1} + \mathbf{x}_{t,2} + \tilde{\mathbf{c}}_f \right)}{\left| \sum_{t \in f} \Omega_t \left(\mathbf{x}_{t,1} + \mathbf{x}_{t,2} + \tilde{\mathbf{c}}_f \right) \right|} \tag{75}$$

where $t \in f$ denotes the spherical triangles t of face f, Ω_t is spherical triangle's solid angle which is calculated using l'Huilier's theorem, $\mathbf{x}_{t,1}$ and $\mathbf{x}_{t,2}$ are the positions of the vertices shared by the face f and spherical triangle t, and $\tilde{\mathbf{c}}_f$ is the position of the centre vertex shared by all spherical triangles of face f. The face centre radius r is the mean radius of the face vertices, again weighted by the solid angle of each spherical triangle,

$$r = \frac{\sum_{t \in f} \Omega_t \left(\left| \mathbf{x}_{t,1} \right| + \left| \mathbf{x}_{t,2} \right| \right) / 2}{\Omega_f} \tag{76}$$

where the solid angle Ω_f of face f is the sum of the solid angles of the constituent spherical triangles,

$$\Omega_f = \sum_{t \in f} \Omega_t \tag{77}$$

We use equations (75) and (76) to calculate the centre $\mathbf{c}_f$ of the face,

$$\mathbf{c}_f = r\,\hat{\mathbf{r}} \tag{78}$$

The area vector $\mathbf{S}_f$ of the surface face f is the sum of the spherical triangle areas [39],

$$\mathbf{S}_f = r^2 \Omega_f \hat{\mathbf{r}} \tag{79}$$

Cell centres and cell volumes are corrected by considering faces that are not normal to the sphere such that

$$\frac{\left(\mathbf{S}_f \cdot \mathbf{c}_f\right)^2}{\left|\mathbf{S}_f\right|^2 \left|\mathbf{c}_f\right|^2} > 0 \tag{80}$$

TODO: *in this case we use a more general method for identifying surface faces than we used for face correction. we should use the general method in all cases and update this description accordingly. raised* `https://trello.com/c/anBhIy60/812-change-cell-centre-spherical-correction-to-be-consistent-with-edge-and-face-centres` *for this.* Let $\mathcal{F}$ be the set of faces satisfying equation (80). Then, the cell volume $\mathcal{V}_c$ is

$$\mathcal{V}_c = \frac{1}{3} \sum_{f \in \mathcal{F}} \mathbf{S}_f \cdot \mathbf{c}_f \tag{81}$$

which can be thought of as the area A integrated between r_1 and r_2 such that $\int_0^R A(r)\,\mathrm{d}r = \int_{r_1}^{r_2} r^2 \Omega\,\mathrm{d}r = \frac{1}{3}\Omega\left(r_2^3 - r_1^3\right)$.
TODO: *this will be amended in due course — The cell centre position* $\mathbf{c}_c$ *is*

$$\mathbf{c}_c = \frac{\sqrt{\frac{1}{3}\left(r_1^2 + r_1 r_2 + r_2^2\right)} \sum_{f \in \mathcal{F}} \mathbf{c}_f}{\left|\sum_{f \in \mathcal{F}} \mathbf{c}_f\right|} \tag{82}$$

Edges can be classified in a similar manner to faces where surface edges are tangent to the sphere and radial faces are normal to the sphere. The edge midpoints, $\mathbf{c}_e$, are used to calculate the face flux for non-divergent velocity fields (equation 73). For transport tests, corrections to edge midpoints are unnecessary. Due to the choice of r_1 and r_2 during mesh construction, the midpoint of a radial edge is at a radial distance of R_e which is necessary for the correct calculation of non-divergent velocity fields. The position of surface edge midpoints is unimportant because these edges do not contribute to the face flux since $\mathbf{e} \cdot \mathbf{c}_e = 0$. Edge lengths are the straight-line distance between the two vertices and not the great-circle distance. Again, the edge lengths are not corrected because it makes no difference to the face flux calculation.

[1] R. L. Walko, R. Avissar, The Ocean-Land-Atmosphere Model (OLAM). Part II: Formulation and tests of the nonhydrostatic dynamic core, Mon. Wea. Rev. 136 (2008) 4045–4062. doi:10.1175/2008MWR2523.1.

[2] C. Schär, D. Leuenberger, O. Fuhrer, D. Lüthi, C. Girard, A new terrain-following vertical coordinate formulation for atmospheric prediction models, Mon. Wea. Rev. 130 (2002) 2459–2480. doi:10.1175/1520-0493(2002)130<2459:ANTFVC>2.0.CO;2.

[3] K. P. Hoinka, G. Zängl, The influence of the vertical coordinate on simulations of a pv streamer crossing the alps, Mon. Wea. Rev. 132 (7) (2004) 1860–1867. doi:10.1175/1520-0493(2004)132<1860:TIOTVC>2.0.CO;2.

[4] S. Webster, A. Brown, D. Cameron, C. Jones, Improvements to the representation of orography in the Met Office Unified Model, Quart. J. Roy. Meteor. Soc. 129 (2003) 1989–2010. doi:10.1256/qj.02.133.

[5] T. Gal-Chen, R. C. Somerville, On the use of a coordinate transformation for the solution of the Navier-Stokes equations, J. Comp. Phys. 17 (1975) 209–228. doi:10.1016/0021-9991(75)90037-6.

[6] J. B. Klemp, A terrain-following coordinate with smoothed coordinate surfaces, Mon. Wea. Rev. 139 (2011) 2163–2169. doi:10.1175/MWR-D-10-05046.1.

[7] S. D. Eckermann, J. P. McCormack, J. Ma, T. F. Hogan, K. A. Zawdie, Stratospheric analysis and forecast errors using hybrid and sigma coordinates, Mon. Wea. Rev. 142 (1) (2014) 476–485. doi:10.1175/MWR-D-13-00203.1.

[8] A. J. Simmons, D. M. Burridge, An energy and angular-momentum conserving vertical finite-difference scheme and hybrid vertical coordinates, Mon. Wea. Rev. 109 (1981) 758–766. doi:10.1175/1520-0493(1981)109<0758:AEAAMC>2.0.CO;2.

[9] D. Leuenberger, M. Koller, O. Fuhrer, C. Schär, A generalization of the SLEVE vertical coordinate, Mon. Wea. Rev. 138 (2010) 3683–3689. doi:10.1175/2010MWR3307.1.

[10] H. Yamazaki, T. Satomura, N. Nikiforakis, Three-dimensional cut-cell modelling for high-resolution atmospheric simulations, Quart. J. Roy. Meteor. Soc. 142 (2016) 1335–1350. doi:10.1002/qj.2736.

[11] J. Steppeler, H.-W. Bitzer, M. Minotte, L. Bonaventura, Nonhydrostatic atmospheric modeling using a z-coordinate representation, Mon. Wea. Rev. 130 (2002) 2143–2149. doi:10.1175/1520-0493(2002)130<2143:NAMUAZ>2.0.CO;2.

[12] S. Jebens, O. Knoth, R. Weiner, Partially implicit peer methods for the compressible Euler equations, J. Comp. Phys. 230 (2011) 4955–4974. doi:10.1016/j.jcp.2011.03.015.

[13] J. Shaw, H. Weller, Comparison of terrain following and cut cell grids using a non-hydrostatic model, Mon. Wea. Rev. 144 (2016) 2085–2099. doi:10.1175/MWR-D-15-0226.1.

[14] B. Leonard, M. MacVean, A. Lock, Positivity-preserving numerical schemes for multidimensional advection, Tech. Rep. 106055, NASA (1993).

[15] B. Leonard, A. Lock, M. MacVean, Conservative explicit unrestricted-time-step multidimensional constancy-preserving advection schemes, Mon. Wea. Rev. 124 (11) (1996) 2588–2606. doi:10.1175/1520-0493(1996)124<2588:CEUTSM>2.0.CO;2.

[16] Y. Chen, H. Weller, S. Pring, J. Shaw, Dimension splitting errors and a long time-step multi-dimensional scheme for atmospheric transport, Quart. J. Roy. Meteor. Soc.In prep.

[17] S.-J. Lin, R. B. Rood, Multidimensional flux-form semi-Lagrangian transport schemes, Mon. Wea. Rev. 124 (9) (1996) 2046–2070. doi:10.1175/1520-0493(1996)124<2046:MFFSLT>2.0.CO;2.

[18] K. K. Katta, R. D. Nair, V. Kumar, High-order finite-volume transport on the cubed sphere: Comparison between 1D and 2D reconstruction schemes, Mon. Wea. Rev. 143 (7) (2015) 2937–2954. doi:10.1175/MWR-D-13-00176.1.

[19] A. Iske, M. Käser, Conservative semi-Lagrangian advection on adaptive unstructured meshes, Numer. Methods. Partial Differ. Equ. 20 (3) (2004) 388–411. doi:10.1002/num.10100.

[20] P. H. Lauritzen, R. D. Nair, P. A. Ullrich, A conservative semi-lagrangian multi-tracer transport scheme (cslam) on the cubed-sphere grid, J. Comp. Phys. 229 (5) (2010) 1401–1424.

[21] P. H. Lauritzen, C. Jablonowski, M. A. Taylor, R. D. Nair, Numerical techniques for global atmospheric models, Vol. 80, Springer Science & Business Media, 2011. doi:10.1007/978-3-642-11640-7.

[22] R. K. Lashley, Automatic generation of accurate advection schemes on unstructured grids and their application to meteorological problems, Ph.D. thesis, University of Reading (2002).

[23] W. C. Skamarock, M. Menchaca, Conservative transport schemes for spherical geodesic grids: High-order reconstructions for forward-in-time schemes, Mon. Wea. Rev. 138 (12) (2010) 4497–4508. doi:10.1175/2010MWR3390.1.

[24] P. H. Lauritzen, C. Erath, R. Mittal, On simplifying 'incremental remap'–based transport schemes, J. Comp. Phys. 230 (22) (2011) 7957–7963. doi:10.1016/j.jcp.2011.06.030.

[25] J. Thuburn, C. Cotter, T. Dubos, A mimetic, semi-implicit, forward-in-time, finite volume shallow water model: comparison of hexagonal-icosahedral and cubed-sphere grids, Geosci. Model Dev. 7 (2014) 909–929. doi:10.5194/gmd-7-909-2014.

[26] H. Weller, H. G. Weller, A. Fournier, Voronoi, Delaunay, and block-structured mesh refinement for solution of the shallow-water equations on the sphere, Mon. Wea. Rev. 137 (12) (2009) 4208–4224. doi:10.1175/2009MWR2917.1.

[27] W. C. Skamarock, A. Gassmann, Conservative transport schemes for spherical geodesic grids: High-order flux operators for ODE-based time integration, Mon. Wea. Rev. 139 (2011) 2962–2975. doi:10.1175/MWR-D-10-05056.1.

[28] P. Lauritzen, P. Ullrich, C. Jablonowski, P. Bosler, D. Calhoun, A. Conley, T. Enomoto, L. Dong, S. Dubey, O. Guba, et al., A standard test case suite for two-dimensional linear transport on the sphere: results from a collection of state-of-the-art schemes, Geosci. Model Dev. 7 (2014) 105–145. doi:10.5194/gmd-7-105-2014.

[29] H. Miura, An upwind-biased conservative advection scheme for spherical hexagonal-pentagonal grids, Mon. Wea. Rev. 135 (12) (2007) 4038–4044. doi:10.1175/2007MWR2101.1.

[30] K.-S. Yeh, The streamline subgrid integration method: I. quasi-monotonic second-order transport schemes, J. Comp. Phys. 225 (2) (2007) 1632–1652. doi:10.1016/j.jcp.2007.02.033.

[31] A. Gassmann, A global hexagonal C-grid non-hydrostatic dynamical core (ICON-IAP) designed for energetic consistency, Quart. J. Roy. Meteor. Soc. 139 (670) (2013) 152–175. doi:10.1002/qj.1960.

[32] H. Weller, A. Shahrokhi, Curl free pressure gradients over orography in a solution of the fully compressible Euler equations with implicit treatment of acoustic and gravity waves, Mon. Wea. Rev. 142 (2014) 4439–4457. doi:10.1175/MWR-D-14-00054.1.

[33] P. H. Lauritzen, W. C. Skamarock, M. Prather, M. Taylor, A standard test case suite for two-dimensional linear transport on the sphere, Geosci. Model Dev. 5 (2012) 887–901. doi:10.5194/gmd-5-887-2012.

[34] CFD Direct, OpenFOAM user guide: Numerical schemes, http://cfd.direct/openfoam/user-guide/fvschemes/ (2016).

[35] B. Good, A. Gadian, S.-J. Lock, A. Ross, Performance of the cut-cell method of representing orography in idealized simulations, Atmos. Sci. Lett. 15 (2014) 44–49. doi:10.1002/asl2.465.

[36] M. Jähn, O. Knoth, M. König, U. Vogelsberg, ASAM v2.7: a compressible atmospheric model with a Cartesian cut cell approach, Geosci. Model Dev. 8 (2) (2015) 317–340. doi:10.5194/gmd-8-317-2015.

[37] U. Vogelsberg, ASAM wiki: Grid generator, http://asamwiki.tropos.de/index.php?title=Grid_Generator (2010).

[38] A. Staniforth, J. Thuburn, Horizontal grids for global weather and climate prediction models: a review, Quart. J. Roy. Meteor. Soc. 138 (2012) 1–26. doi:10.1002/qj.958.

[39] G. Van Brummelen, Heavenly mathematics: The forgotten art of spherical trigonometry, Princeton University Press, 2013.

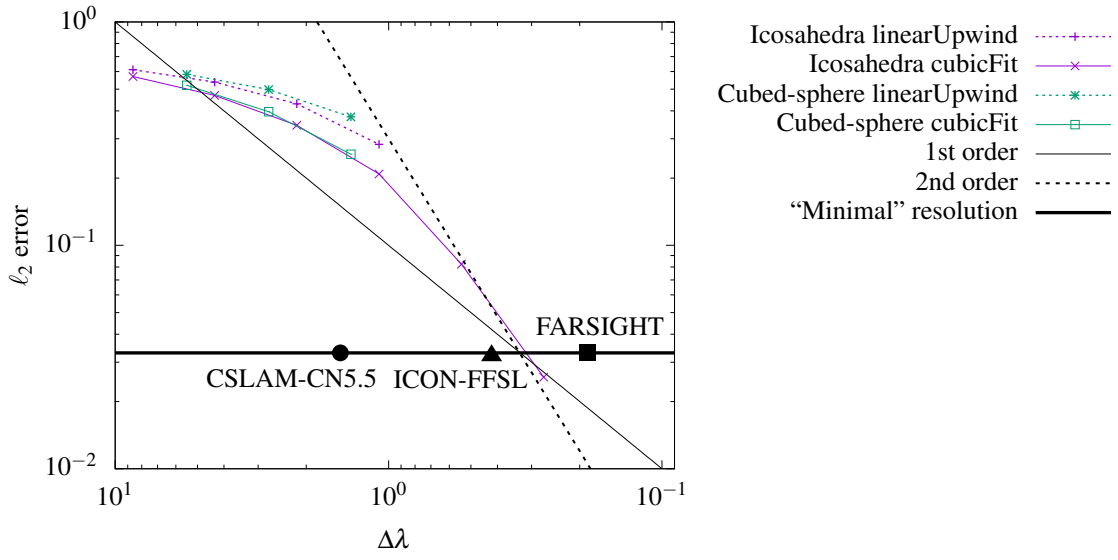

Figure 13: TODO: *remove this figure* TODO: *ℓ_2 convergence for non-divergent deformational flow using Cosine bells. Used to find "minimal" resolution.*