University of Reading
Department of Mathematics

The Semi-Lagrangian Method in Atmospheric Modelling

Christopher J. Smith

Thesis submitted for the degree of Doctor of Philosophy.

March 2000

Contents

List of Figures

= 0p tsmlaal egon

List of Tables

Introduction

For much of its history, the modern science of meteorology has been dominated by weather forecasting [41]. Even slight gains in forecasting accuracy are of great benefit to areas such as aviation, shipping and farming. Forecasting now commands the use of extensive ground-based and satellite data gathering networks. Weather simulations are run on the largest and most powerful computers. Newer uses of atmospheric simulation range from the long term predictions of climate modelling, to the modelling of local environmental e□ects of weather. Climate models have an impact on long-term policy making at a national and international level. While at the local level, the contribution of atmospheric modelling to areas such as pollution dispersion, building design and flood and storm warnings is just as great. The uses for atmospheric modelling increase to keep pace with this rapidly developing area of research.

Many physical processes may be represented in atmospheric models. Climate prediction requires the coupling together of many physical systems. Important considerations in this respect are, for instance, the coupling between the oceans and atmosphere; the interaction between cloud formation and the global energy b

Chapter 1

The Semi-Lagrangian Method

1.1 Introduction

Mathematical descriptions of fluid flow divide into two groups: Eulerian and Lagrangian. Eulerian descriptions work within a single fixed frame of reference, through which the fluid flows. This is rather like observing the flow of a river while stood on one of its banks. The second manner of description is that from within the fluid. This second kind of formalism is known as the Lagrangensm

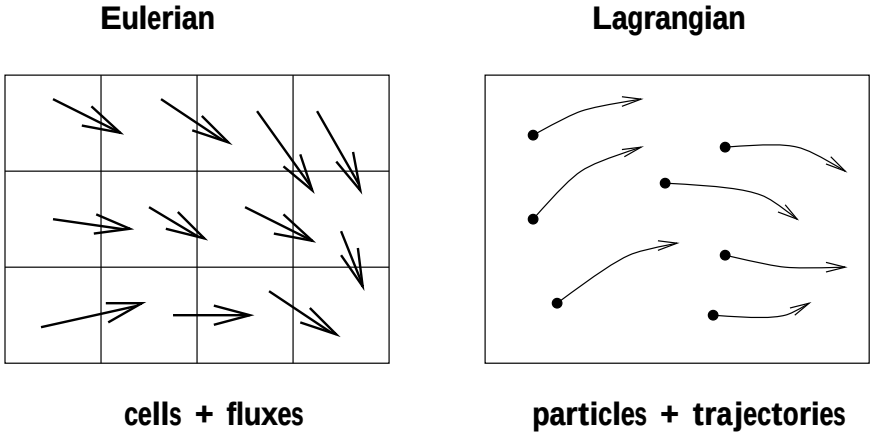


Figure 1.1: Eulerian and Lagrangian numerical models

model, by contrast, does not represent the fluid in terms of continuum ideas at all. The two basic constructions within the Lagrangian model are individual particles and their trajectories. Each particle carries values for all the fluid properties, and moves with the velocity of the fluid flow. Semi-Lagrangian models may be viewed as a hybrid of

based Eulerian scheme is the difficulty of achieving stable and accurate simulations with long time-steps. For time-steps sufficiently long to allow fluid to move across several cells in a single step, the sequence of flux transfers between cells becomes extremely difficult to represent [22]. In practice, schemes of this form are operated such that only flows between

linear advection problems.

1.2 Linear Advection using SL

In this section we consider how the semi-Lagrangian method may be used to solve linear advection equations in one dimension. The problems to be solved are of the following general form:

$$\begin{aligned} \frac{\partial u}{\partial t} + a(x,t)\frac{\partial u}{\partial x} &= g(x,t) \\ u(x,0) &= u_0(x). \end{aligned} \tag{1.1}$$

We shall develop a SL scheme for [illegible] two steps. First we restrict our

[illegible]

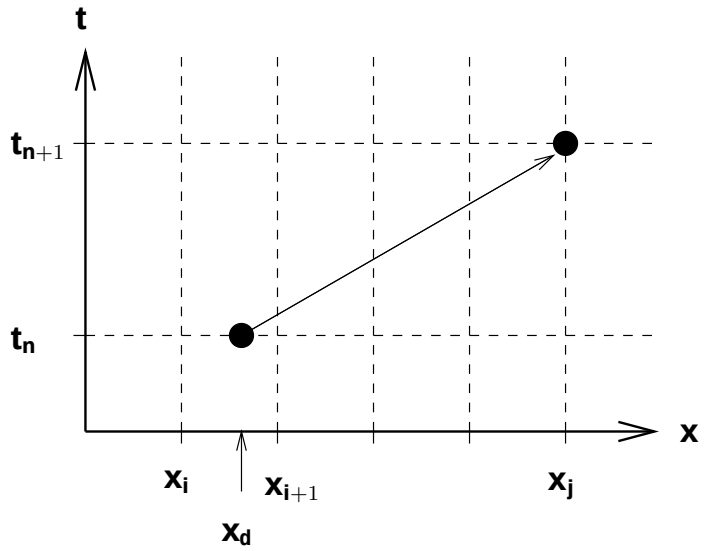


Figure 1.2: Basic SL method

which, starting at time t_n, arrives at the grid-point x_j at the later time t_{n+1}. How this is done for a general velocity field is considered in the next section.

To continue with the development of the SL scheme, we next make use of the analytical solution which states that u remains constant on any characteristic. For more general advection problems it is at this point where suitable time discretisations must be considered. These are discussed in section 1.2.3 and Chapter 11. For now, we use this property of characteristics to obtain

$$u_j^{n+1} = u_d^n, \tag{1.5}$$

where u_d^n represents the value of u at the departure point. This equation provides the finite difference solution at the new time-level, from the data at the old level. There are now two possibilities. First, if x_d happens to coincide with a grid-point, say $x_d = x_i$, then (1.5) provides the scheme

$$u_j^{n+1} = u_i^n.$$

Generally the departure points will not be coincident with grid-points. This is where interpolation is required. For this purpose polynomial interpolation may be used. For instance, if x_d lies between grid-points x_i and x_{i+1}, then linear interpolation of the finite

di□erence data at these points provides

$$u_d^n = \left(\frac{x_d - x_i}{x_{i+1} - x_i}\right) u_{i+1}^n + \left(\frac{x_{i+1} - x_d}{x_{i+1} - x_i}\right) u_i^n,$$

where u_d^n is now only an approximation to $u(x_d, t_n)$. Using once again the constancy of u along characteristics, now coupled with the above interpol

The implicit mid-point rule is based on a discretisation of t

Simple extrapolation formulae may be obtained by considering Taylor series expansions. We mention here the two most practical formulae obtained in this way:

- **2nd order accurate: $a^{n+1/2} = {}^{1}$**

values of θ provide semi-implicit discretisations. Since these schemes only involve two time-levels, they give results which are at best second order accurate. Greater accuracy may be obtained by increasing the number of time-levels used. Such schemes are discussed in Chapter 11.

A solution procedure is required for obtaining u_j^{n+1} from (1.7). This will depend on the nature of the source terms. Explicit terms are evaluated by interpolation of the existing finite difference data. Implicit terms, however, may involve other fluid variables. Consequently there will be a system of discretisations of the form (1.7), one for each fluid variable. These together with whatever other equations are required to represent the fluid system are to be solved simultaneously for all fluid variables. This is the case, for instance, when solving the shallow water equations, as discussed in Chapter 8.

Interpolation is a procedure for increasing the extent of a given amount of data. This can only be achieved by making assumptions about the quantity represented by this data. For this reason it is not surprising that the quality of any SL scheme depends strongly on the chosen interpolation method. In the following two chapters we examine the development of interpolation methods suitable for SL advection.

1.3 Nonlinear Advection

Semi-Lagrangian methods are not restricted solely to linear advection problems. Central to any model of fluid flow is a nonlinear advection equation representing the transport of momentum.

The appropriate momentum equation in atmospheric simulation is the Navier-Stokes equation. Models which are used in practice are built around some approximation of this equation. In addition to a 92352(o)-2.26309(m)2.92352(e)3.56312(n)32.0607(t)-0.6481742.51097(a)3

model. The non-hydrostatic model of Tapp and White [58] uses the equation set

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}.\nabla)\mathbf{u} = \mathcal{F} + \mathbf{g} - \frac{1}{\rho}\nabla p \quad \text{momentum}$$

$$\frac{\partial \rho}{\partial t} + \mathbf{u}.\nabla\rho = -\rho\nabla.\mathbf{u} \quad \text{continuity of mass}$$

$$\frac{\partial c_p T}{\partial t} + \mathbf{u}.\nabla(c_p T) = Q \quad \text{conservation of heat}$$

$$\rho = \frac{p M_r}{RT}$$

where a is a label for individual particles. A suitable label is provided by a particle's position at some reference time, as that uniquely specifies a particle for all subsequent times. In order to distinguish time derivatives in the Lagrangian coordinate system from those in the Eulerian system, the notation $\frac{D}{Dt}$ is often used in place of $\frac{d}{dt}$. Since the transformation between the two coordinate systems is simply translation with the fluid velocity u, the following identity holds

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + u.\nabla.$$

In the Lagrangian system the particle labeled a has position $x(a,t)$ at time t. Hence particle position is a dependent variable of the Lagrangian model in addition to those listed for the Eulerian model. As such it must also have an evolution equation. This is just the trajectory equation, that states that each fluid particle moves with the local fluid velocity:

$$\frac{dx}{dt} = u(x(t),t).$$

The other equations of the flow are

$$\frac{Du}{Dt} = \mathcal{F} + g - \frac{1}{\rho}\nabla p$$

$$\frac{D\rho}{Dt} = -\rho\nabla.u$$

$$c_p\frac{DT}{Dt} = Q,$$

and the equation of state is as before. When the SL method is applied to such a system of equations, the trajectory equation is first used to determine departure points. Then the

e mit mmov589(u)1..9482(e)-277.487(S)1.191(s)4.7933(t)-0.64694u4.128(n)2.9235-0.64694h(l)0.971648

water equations are just such a set, and take the form

$$\frac{Du}{Dt} = -g\frac{\partial h}{\partial x}$$

$$\frac{Dv}{Dt} = -g\frac{\partial h}{\partial y}$$

$$\frac{Dh}{Dt} = -h\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right).$$

The three model variables are u and v, the two-dimensional velocity components, and h the fluid depth; g is the gravitational constant. This two-dimensional model

Chapter 2

Interpolation: One-Dimensional Schemes

2.1 Introduction

In this chapter we begin the description of how multi-dimensional semi-Lagrangian methods are constructed. Any practical weather prediction simulation requires the modelling of fully three-dimensional flows. The three aspects of the semi-Lagrangian method outlined in the first chapter each apply in any number of space dimensions. No new theoretical considerations arise for the SL method as a whole. The task is one of finding suitable algorithms for each of the three stages: trajectory calculation, interpolation and time

interpolation in one dimension.

2.2 Univariate Interpolation

Multivariate interpolation is typically carried out by combining several univariate interpolations. Tensor product interpolation provides the most familiar example of this approach, for which details are given in Chapter 5. A further method of this kind has been developed recently by Purser and Lesley [39], [40], [23]. In what they call "cascade interpolation" data is transferred e□ciently from one grid to another. Such an emphasis on transferring whole representations of multivariate functions makes this method particularly suitable for SL. Cascade interpolation is also described in Chapter 5. Before considering these methods we must first examine the options for interpolation in one-dimension, from which multi-dimensional methods In choosing an interpolation method, the ultimate end for which it is to be used dictates the constraints to be satisfied by the interpolant. For instance, if we know that the data to be interpolated represent a quantity which is everywhere non-negative, such as atmospheric water content, then there will be an advantage in finding an interpolant which respects this property. This leads us to consider how the degrees of freedom possessed by an interpol

The interpolant $p(x)$ is a function with a finite number of degrees of freedom. If p has M degrees of freedom, then we must impose M conditions on p for it to be determined completely. Since $p(x)$ is required to be interpolant for the set S it must be consistent with S. That is, p must be such that $M \geq N$, and it must satisfy the interpolation conditions

$$p(x_i) = f_i \quad i = 0, \cdots, N \tag{2.1}$$

If it is the case that $M = N + 1$ then (2.1) determine the interpolant completely. However, only piecewise constant functions fit this description. For most practical applications smoother interpolation is required, and we are led to consider interpolants for which $M > N + 1$. In addition to the $N + 1$ interpolation conditions, a further $M - N - 1$ conditions must be specified for such an interpolant, and we are free to choose what these should be. For example, consider piecewise interpolation using cubic polynomials.

2.2.1 Example: Piecewise Cubic Interpolation

On each sub-interval $[x_i, x_{i+1}]$ the interpolant is a cubic polynomial,

$$p(x) = p_i(x) = A_i x^3 + B_i x^2 + C_i x + D_i.$$

Since there are N such sub-intervals and each cubic has four degrees of freedom the interpolant has $M = 4N$ degrees of freedom. We can match conditions to these degrees of freedom in various ways. Piecewise cubic Lagrange interpolation is obtained by requiring $p_i(x)$ to interpolate the four points $\{(x_{i+k}, f_{i+k}) : k = -1, 2\}$, which surround its interval of definition. To ensure the interpolants in the first and last sub-intervals also satisfy four such conditions, two extra data points are required, one beyond each end of the interval $[a, b]$.

It is not necessary for all four degrees of freedom in each cubic to be assigned through Lagrange interpolation. In order for the interpolant to satisfy the conditions (2.1) and be continuous, each cubic $p_i(x)$ need only interpolate the data points at the two ends of its sub-interval, $[x_i, x_{i+1}]$. This accounts for $2N$ degrees of freedom, half the number which

must be solved to find the departure point x_d corresponding to each grid point x_j. Since the velocity is constant this is a simple matter, and the analytical solution may be used:

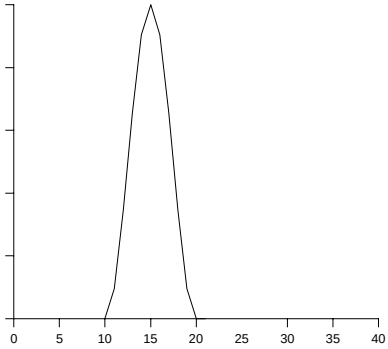
$$x_d = x_j - a\Delta t. \tag{2.4}$$

It is important to note that, for this particular test problem, the time discretisation behaves exactly as the analytical solution: along particle trajectories, u remains constant. The only part of the SL scheme in this case, therefore, which is not exact is the interpolation required to approximate $u^n(x_d)$. Smolarkiewicz and Grell [52] have provided a formal framework in which numerical advection and interpolation are seen to be equivalent operations. We shall return to this in Chapter 4.

Before applying piecewise linear interpolation to the test-problem, we first non-dimensionalise the displacement represented in (2.4). For this problem the dimensionless displacement is identical to the dimensionless velocity, or Courant number,

$$\nu = a\frac{\Delta t}{\Delta x} = \frac{x_j - x_d}{\Delta x}.$$

The integer part of ν represents the number of whole cells traversed in one time-step. If ν happens to be an integer then we may use $u^n(x_d) = u^n_{j-\nu}$, and the resulting fully discrete


0
5
10
15
20
25
30
35
40

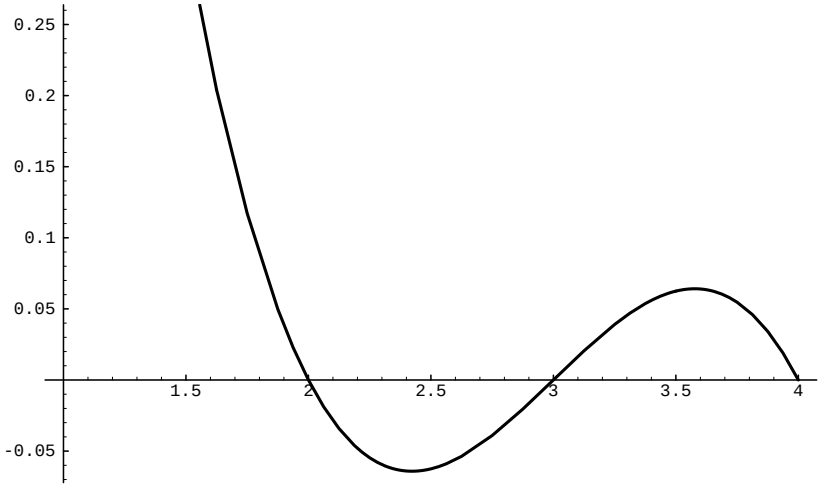


Figure 2.2: Lagrange basis function l_1

a square pulse, using cubic interpolation; equally apparent is the loss of monotonicity in the numerical solution.

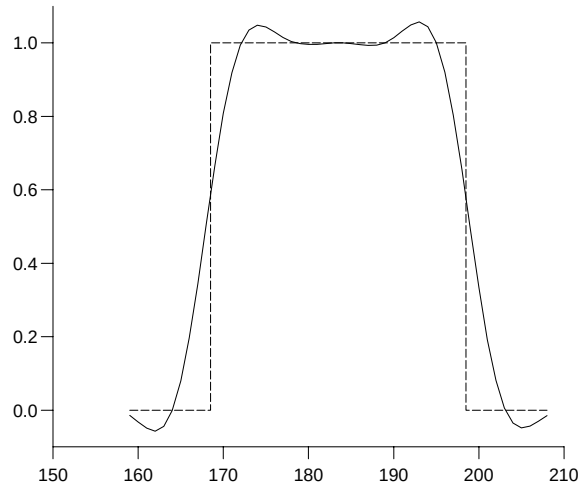


Figure 2.4: High order SL using cubic interpolation

Divided Difference Interpolation

In practice, interpolating polynomials are often constructed using the method of divided differences. This approach allows the order of the interpolant to be increased through successively adding further terms to the interpolant. Divided differences for the data set S are defined in the following way:

$$f[x_i] = f_i$$

$$f[x_i, x_{i+1}, \cdots, x_{i+j}] = \frac{f[x_{i+1}, \cdots, x_{i+j}] - f[x_i, \cdots, x_{i+j-1}]}{x_{i+j} - x_i}.$$

The degree n interpolating polynomial through the points $\{(x_{i+j}, f_{i+j}) : j = 0, \cdots, n\}$ is

$$\begin{aligned} p(x) = f[x_i] &+ (x - x_i) f[x_i, x_{i+1}] \\ &+ (x - x_i)(x - x_{i+1}) f[x_i, x_{i+1}, x_{i+2}] \\ &+ \cdots + (x - x_i) \cdots (x - x_{i+p}) f[x_i, \cdots, x_{i+n}]. \end{aligned}$$

This polynomial is identical to the Lagrange form (2.6), mer

where $f_i = f(x_i)$ and $d_i = f'(x_i)$. The polynomial may be constructed from basis functions, in much the same way as for Lagrange interpolation [15]:

$$p(x) = f_i H_1(x) + f_{i+1} H_2(x) + d_i H_3(x) + d_{i+1} H_4(x),$$

where

$$\begin{aligned}
H_1(x) &= \frac{(x_{i+1} - x)^2}{h^2} + 2\frac{(x - x_i)(x_{i+1} - x)^2}{h^3}, \\
H_2(x) &= \frac{(x - x_i)^2}{h^2} + 2\frac{(x_{i+1} - x)(x - x_i)^2}{h^3}, \\
H_3(x) &= \frac{(x - x_i)(x_{i+1} - x)^2}{h^2}, \\
H_4(x) &= -\frac{(x_{i+1} - x)(x - x_i)^2}{h^2},
\end{aligned}$$

and $h = x_{i+1} - x_i$.

To obtain a complete SL scheme using Hermite interpolation we require some means of estimating the derivative values, d_i and d_{i+1}. At this point we appear to be no fur-

For general initial data the solutions at di□erent times are related through the evolution operator,

- **Evolution operator:** $u(x, t + \Delta t) = E\, u(x, t) \equiv u(x - a\Delta t, t)$
- **Fourier symbol:** $F = e^{-iv\varphi}$.

We next demonstrate how equivalent properties are found for a numerical scheme, using linear interpolation as an example. Let S be the spatial shift operator, defined by

$$S\, u_i = u_{i+1}.$$

In a SL scheme the finite di□erence stencil adapts to the trajectory of the flow. To represent this mechanism in the analysis, the Courant numbe

linear stability. Next considering the complex argument of F_1 we again see that the integer part of the Courant number is rendered exactly by the numerical scheme:

$$\begin{aligned} \arg(F_1) &= \arg[e^{-il\varphi}(1 - \alpha + \alpha e^{-i\varphi})] \\ &= \arg(e^{-il\varphi}) + \arg(1 - \alpha + \alpha e^{-i\varphi}) \\ &= -l\varphi + \arg(1 - \alpha + \alpha e^{-i\varphi}). \end{aligned}$$

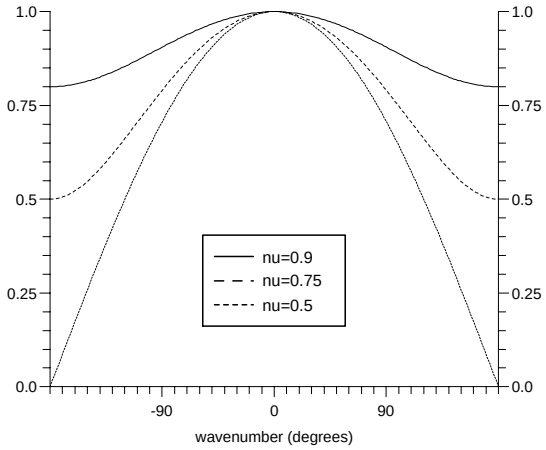
The last term in the above is an approximation relating to the fractional part of the Courant number:

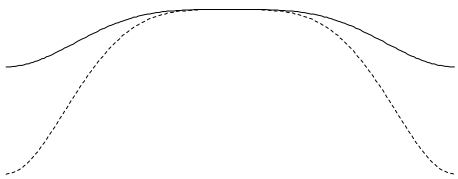
$$\arg(1 - \alpha + \alpha e^{-i\varphi}) = -\alpha\varphi + O(\alpha\varphi^3).$$

For any linear SL scheme on a regular grid the Fourier symbol has a factor $\exp(-il\varphi)$, due to the shift in the stencil. Consequently, as demonstrated above, a complete analysis need only consider Courant numbers in the unit interval, $0 \leq \nu \leq 1$.

Once the Fourier symbol, F_s, of a scheme has been found, we may define two error measurements:

- Amplitude error: ϵ_a =


1.0
0.75
0.5
0.25
0.0
-90
0
90
wavenumber (degrees)
nu=0.9
nu=0.75
nu=0.5



Chapter 3

Quadratic Interpolation

Introduction

Semi-Lagrangian schemes are generally constructed with polynomial interpolants of odd order. Linear interpolation gives insu□cient accuracy for most purposes. Quintic interpolation requires considerable extra computational e□ort for diminishing returns, when compared with cubic interpolation. For this reason cubic interpolation is most often used.

3.1 Centred Quadratic Interpolation

Consider a grid of regularly spaced points with mesh interval Δx :

$$x_j = j\Delta x \quad j = 1, \cdots, N. \tag{3.1}$$

Let f_j

in a semi-Lagrangian scheme, must necessarily satisfy only two interpolation conditions

$$q_k(x_k) = f_k \tag{3.4}$$

$$q_k(x_{k+1}) = f_{k+1}. \tag{3.5}$$

We are free to determine the remaining degrees of freedom of the polynomial in any way we choose. It is still necessary, though, to analyse the stability properties of the resulting SL scheme, since interpolation alone doesn't guarantee stability. We next consider three centred quadratic interpolants.

Interpolant 1: Mean of Left and Right Interpolants

For a quadratic polynomial which satisfies 2 [illegible]

This polynomial provides the SL equivalent of Fromm's scheme. The polynomial (3.8) interpolates, as required, to the two central data points at x_k and x_{k+1}. It doesn't interpolate the data at any other grid-points.

For interpolation on an irregular grid, we may still use q^F as a piecewise interpolant.

We next determine the value of C for the quadratic which minimises Q.

Substituting (3.12) into (3.11), we obtain

$$Q = \frac{1}{2}[Ca + l_k(x_{k-1}) - f_{k-1}]^2 + \frac{1}{2}[Cb + l_k(x_{k+2}) - f_{k+2}]^2, \tag{3.14}$$

where

$$a = (x_k - x_{k-1})(x_{k+1} - x_{k-1}), \tag{3.15}$$

$$b = (x_{k+2} - x_k)(x_{k+2} - x_{k+1}). \tag{3.16}$$

Applying the minimisation condition,

$$\frac{dQ}{dC} = 0, \tag{3.17}$$

we obtain

$$C = \frac{a[f_{k-1} - l_k(x_{k-1})] + b[f_{k+2} - l_k(x_{k+2})]}{a^2 + b^2}. \tag{3.18}$$

Interpolant 3: Weighted Least Squares Minimisation

Minimisation of the quantity Q

use in semi-Lagrangian schemes. We begin by examining interpolation error and then SL advection error.

3.2.1 Test 1: Interpolation

We compare the three interpolants q^M, q^Q and q^W, together with the quadratic ENO interpolant q^E, see Section 4.1. The data to be interpolated are given by the function

$$f(x) = \begin{cases} \cos\left[\frac{\pi}{2}(x-1)\right], & 0 \le x < 2 \\ x - 2, & 2 \le x < 3 \\ 4 - x, & 3 \le x < 4 \\ 1, & 4 \le x < 6 \\ \exp[-25(x-7)^2], & 6 \le x \le 8. \end{cases} \tag{3.21}$$

Data obtained with this function are interpolated on K different irregular grids. Let these grids be $x^{(n)}$, where $n = n_1, \cdots, n_1 + K - 1$. Grid $x^{(n)}$ has $n + 1$ grid points, $x_j^{(n)}$, $j = 0, \cdots, n$, where $x_0^{(n)} = 0$ and $x_n^{(n)} = 8$. For the results presented here, we use $n_1 = 24$ and $K = 217$. With this choice of parameters, the first grid has approximately six grid intervals covering each of the four sections in the initial data; the last grid has ten times that number. The interior grid points are given by first calculating y_j, $j = 0, \cdots, n$, where

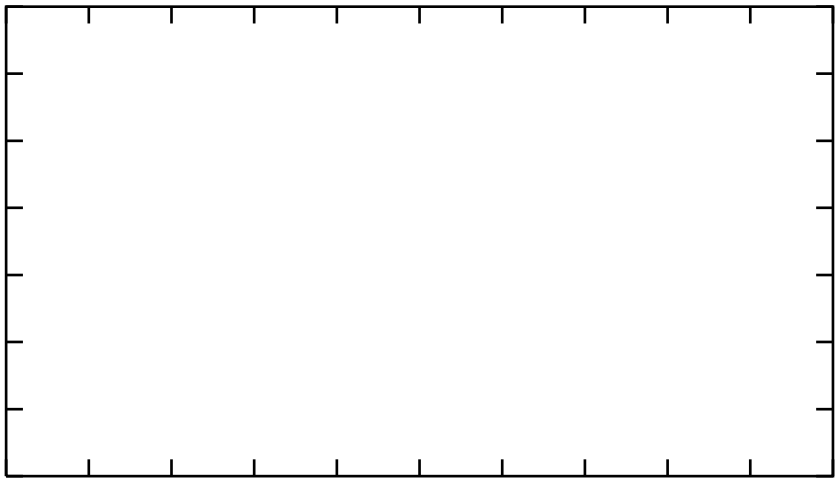
$$\begin{aligned} y_0 &= 0 \\ y_j &= y_{j-1} + 2 + \sin(j), \quad j = 1, \cdots, n. \end{aligned} \tag{3.22}$$

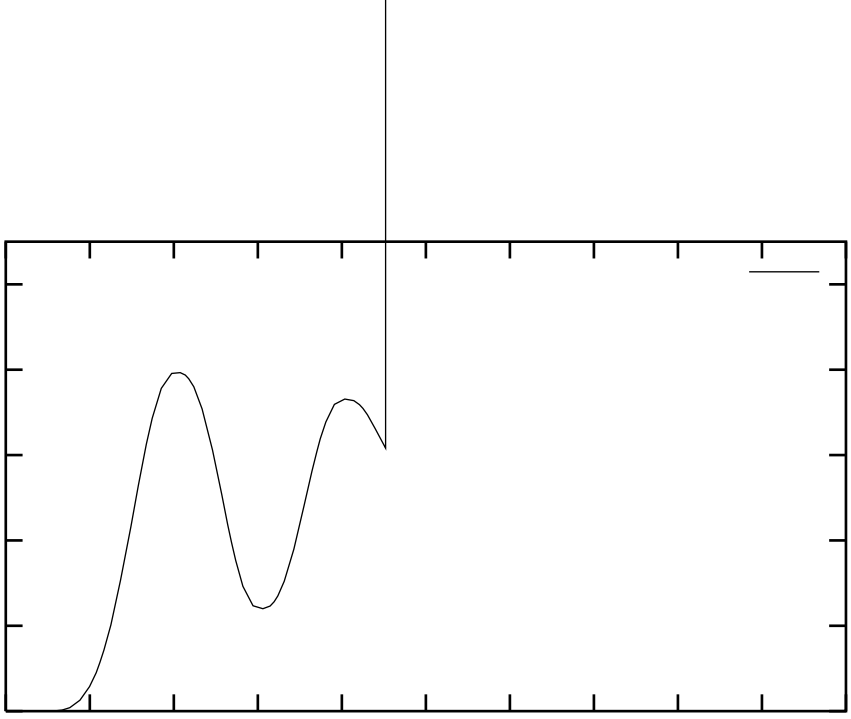
The required grid points are then obtained using

$$x_j^{(n)} = 8\left[\frac{y_j}{y_n}\right], \quad j = 0, \cdots, n. \tag{3.23}$$

For each grid $x^{(n)}$, the data are interpolated to m points, which are equally spaced

At each time step the advected distance is 0.02, which provides a range of local Courant numbers between 0.16 and 0.48. One thousand timesteps are made. The L_2 error in the numerical solution is calculated using only those grid points at which the analytical





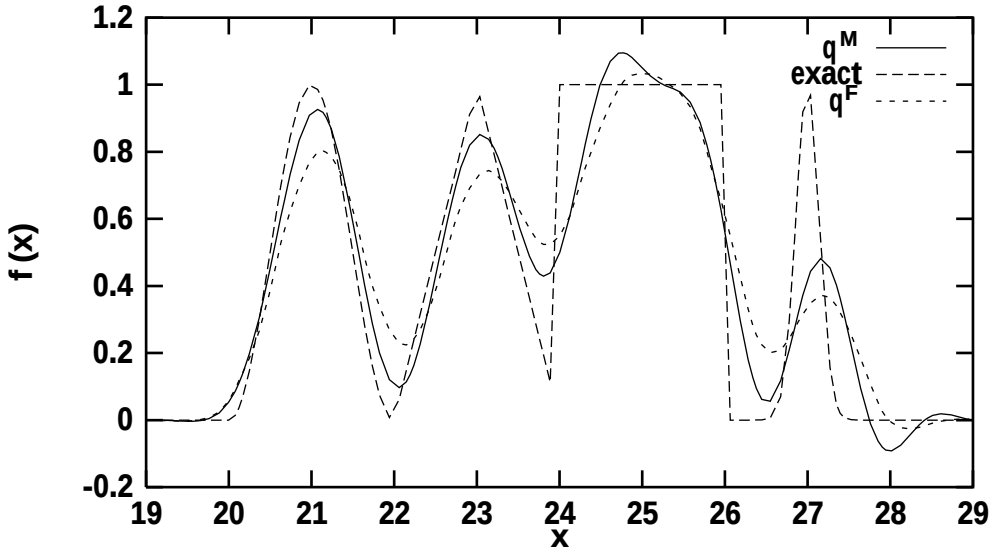


Figure 3.4: Comparison of mean and Fromm quadratic interpolants. Constant speed advection test over an irregular grid. Solutions shown after 1000 timesteps.

3.2.5 Conclusion

The computational tests show that the quadratic Fromm interpolant (3.8) compares well against other, more accurate quadratic interpolants. This scheme has potential for application in semi-Lagrangian advection schemes.

Interpolant (q)	$\mathrm{err}_{\mathrm{L2}}[q]$	min[q]	max[

Chapter 4

Nonlinear Schemes

Linear schemes with high order accuracy introduce spurious small-scale oscillations into the numerical solution, as shown in Figure 2.4. In order to prevent these oscillations nonlinear interpolation methods are required. Three such methods are reviewed here.

4.1 ENO Interpolation

The essentially non-oscillatory (ENO) interpolation employed by Harten, Engquist, Osher and Chakravarthy in their advection schemes [12], [11] is immediately applicable to SL schemes. In this context ENO is a nonlinear adjustment to the Lagrange interpolating polynomial in divided difference form. The following description of ENO interpolation employs the notation of Section 2.4.1.

Starting with linear interpolation, the order of accuracy of the ENO interpolant is increased by the addition of successive terms, producing a sequence of interpolants $q_1, \ldots, q_N$ up to the desired accuracy. Each new term is a polynomial in x and has a divided difference, of appropriate order, as leading coefficient. ENO differs from standard divided difference interpolation through involving a wider range of divided difference values which may be calculated from the data. Using the notation of Section 2.4.1, the

first order ENO interpolant at the point x, where $x_i \le x < x_{i+1}$, is

$$q_1(x) = f[x_i] + (x - x_i) f[x_i, x_{i+1}].$$

Second order ENO interpolation, $q_2(x)$, is formed by adding a quadratic term to the above, of the form

$$f[\ ,\ ,\ ](x - x_i)(x - x_{i+1}),$$

where $f[\ ,\ ,\ ]$ is a second divided difference. The first divided difference, in the linear term, uses data at the two points x_i and x_{i+1}. These same two points occur in two second order divided differences, namely $f[x_{i-1}$

only grow at a rate limited by (4.1). In practice such oscillations remain small compared to those introduced by Lagrange interpolation. Furthermore the oscillations remain localised about steep gradients in the data, and are rapidly damped away from these regions. A second consequence of (4.1) also makes ENO beneficial for us in advection schemes: pre-existing extrema in the data are damped to a lesser degree than they would be by Lagrange interpolation. Unfortunately these first two properties lead to a third which diminishes the suitability of ENO interpolation for some applications. Since ENO allows the depth of a minimum point in the solution to decrease, positivity of interpolation may be lost even when the data are strictly positive. This would make a SL scheme based on ENO interpolation unsuitable for moisture calculations, for instance, unless further steps are taken to preserve positivity.

values to the Hermite cubic at the ends of the interval:

$$d_i \approx f'(x_i)$$

$$d_{i+1} \approx f'(x_i).$$

Any method, such as one of those listed in Section 2.4.1, may be used for the derivative estimates. Once initial estimates for d_i and d_{i+1} have been made, these are then adjusted if necessary to ensure the interpolant remains monotone across the interval. This is done by comparing the estimates against the discrete slope of the data across the interval, $\Delta_i = (f(x_{i+1}) - f(x_i))/(x_{i+1} - x_i)$. If $\Delta_i = 0$ then the interpolation will be monotone across the interval only if both derivative estimates are zero, producing a constant valued interpolation. That is, if $\Delta_i = 0$ then d_i and d_{i+1} are set to zero, regardless of their original values. If Δ_i is non-zero, then a sufficient condition for monotonicity is given in terms of the ratios between derivative estimates and the discrete slope. Let $\alpha = d_i/\Delta_i$ and $\beta = d_{i+1}/\Delta_i$. The interpolation is monotone if the following conditions are satisfied:

$$\begin{cases} 0 \leq \alpha \leq 3 \\ 0 \leq \beta \leq 3. \end{cases}$$

The simplest approach to ensuring these conditions are satisfied, is to reset the derivative estimates according to

$$d_i \leftarrow 0 \quad \text{if } \alpha < 0$$

$$d_i \leftarrow 3\Delta_i \quad \text{if } \alpha > 3.$$

The value of d_{i+1} is adjusted with reference to β in the same way.

order, but monotone, solution to provide a less severe bound where required, we avoid any difficulties in this respect. So, following Priestley, we set

$$\begin{aligned} u_j^+ &= \max\{u_d^{\max}, u_{Lj}^{n+1}\} \\ u_j^- &= \min\{u_d^{\min}, u_{Lj}^{n+1}\}, \end{aligned}$$

and require the QMSL solution to satisfy

$$u_j^- \le u_j^{n+1} \le u_j^+. \tag{4.2}$$

This requirement ensures that no new extrema are created by the interpolation stage of the QMSL scheme. Next we consider how to merge the high and low order solutions to obtain a solution satisfying the constraint (4.2), while preserving the maximum degree of accuracy possible.

Beginning with the low order solution, a correction is sought which will increase the accuracy in the solution,

$$u_j^{n+1} = u_{Lj}^{n+1} + \delta_j^{n+1}.$$

The correction term δ_j^{n+1} is derived from the high order solution. In the following computations the time superscript is dropped for clarity. Let

$$\delta_j = \alpha_j(u_{Hj} - u_{Lj}), \quad 0 \le \alpha_j \le 1,$$

where α_j are to be determined. If all the $\alpha_j = 1$ the QMSL solution is just the high order solution. The task is now to find α_j as large as possible, within the allowed range, such that the monotonicity constraint (4.2) is yet satisfied. This may be achieved by first computing the maximum and minimum allowable corrections,

$$\begin{aligned} Q^+ &= u_j^+ - u_{Lj} \\ Q^- &= u_j^- - u_{Lj}, \end{aligned}$$

and the difference between the two existing solutions,

$$P = u_{Hj} - u_{Lj}.$$

There are now three possibilities:

$$1)\quad P > 0 \;:\; \alpha_j = \min\{1, \frac{Q^+}{P}\}$$

$$2)\quad P < 0 \;:\; \alpha_j = \min\{1, \frac{Q^-}{P}\}$$

$$3)\quad P = 0 \;:\; \alpha_j = 0.$$

numerical solution drops to about 75% of that of the exact solution. The position of the pulse appears to be very close to being exact, though with a slight lag, as anticipated by the Fourier analysis of Section 2.4.2

Repeating the simulation once more, this time using 2327 time-steps to achieve the total displacement of one thousand cells (CFL ≈ 0.43), a quite di□erent result obtains. The peak height has now dropped close to 50% of its exact value. and there is a slightly greater lag in the position of the pulse. In addition, we see that the extent (or support) of the numerical solution has increased considerably: the plots show the portions of the

4xstetos(monot

at the expense of a large error in the value of peak height: the peak height for 423 steps with monotone interpolation is roughly that obtained by 2327 steps with unconstrained interpolation. In terms of the accuracy of the peak value, any gain made by solving at a high Courant number is lost when monotonicity constraints are enforced.

Chapter 5

Interpolation: Multi-Dimensional Schemes

5.1 Introduction

In Chapter 2 we reviewed a selection of methods for univariate interpolation. All of these methods were immediately applicable to SL schemes for one-dimensional problems. Any meaningful approach to numerical weather prediction (NWP) must, however, involve a representation of data throughout a three dimensional domain. Only in such a large model can we hope to describe adequately the processes of the atmosphere. There is a need, therefore, to find multivariate interpolation methods suitable for SL advection.

A simulation using finite differences typically involves a large array of grid points, structured into layers, rows and columns. For each of these grid points a SL scheme would have to compute a departure point. Then for each departure point perform a number of interpolations, one for each model variable. The precise number of variables in a model depends on the number of physical processes to be simulated. A basic weather model would include variables to describe the three components of velocity together with pressure, temperature and moisture. Atmospheric chemistry models may involve many chemical species, each requiring its own model field. Consequently, a SL model may

often involve an extremely large number of interpolations at each time-step. It is readily apparent that much of the computational e□ort in such simulations will be spent on interpolation. The viability of a SL scheme rests on the e□ciency of its interpolation.

A number of methods exist for interpolating multi-dimensional data. Since we require interpolations on a regular grid, we shall restrict our attention to methods suitable for such an arrangement. The most familiar method of this kind is tensor product interpolation,

grids, we can perform a sequence — or cascade — of one dimensional interpolations that transfers data from one to the other. Under certain conditions on the velocity field, the Lagrangian grid surfaces will provide a valid curvilinear coordinate system, (X, Y, Z).

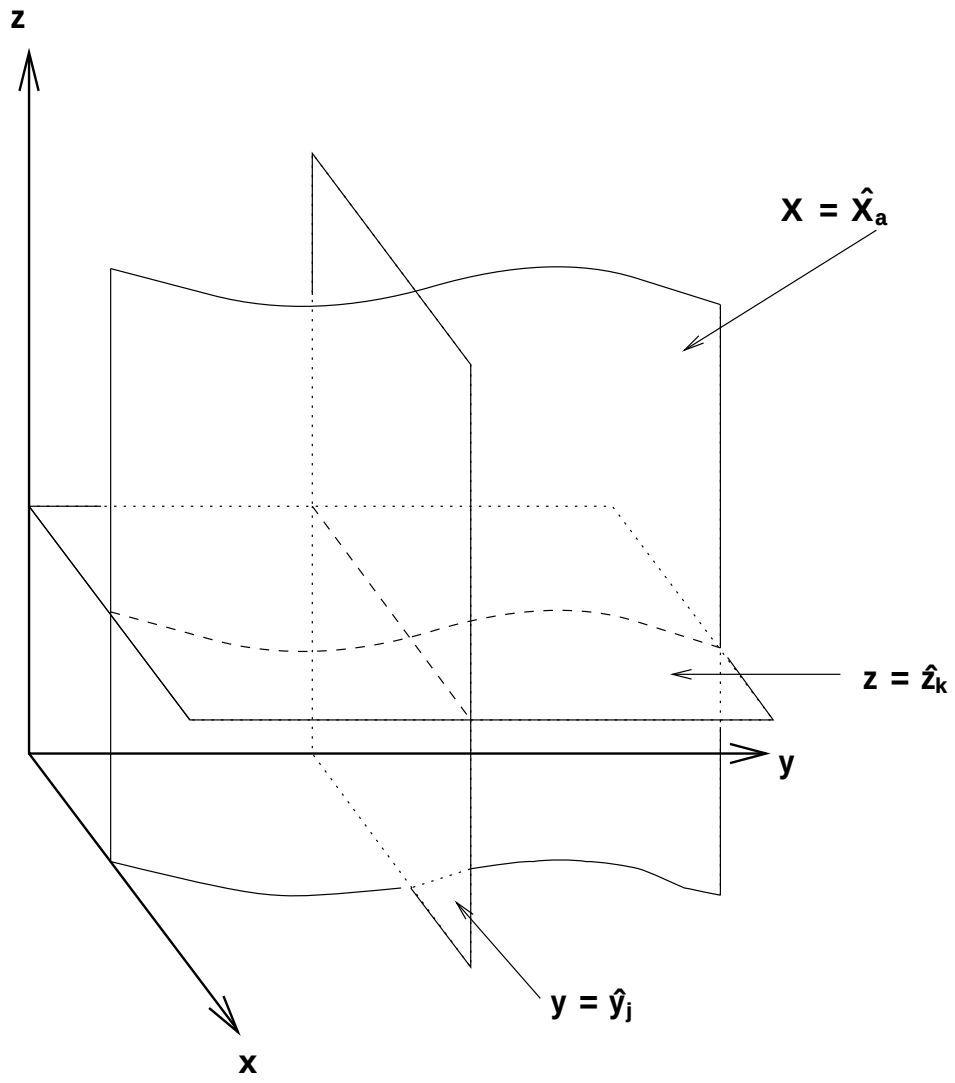


Figure 5.1: Hybrid Grids

The cascade method consists of a sequence of intermediate interpolations, which fproduce approsp(m)2.92352(e)3.56312(d)(a)-2fr352(e)3.56372873(s)4.7933(t)-0.64-488.274(i32(t)-0.6

$\hat{X}_a$ above. So consider the curve of intersection, $(\hat{X}_a, \hat{z}_k)$, between $X = \hat{X}_a$ and $z = \hat{z}_k$. This curve is cut by the plane $Y = \hat{Y}_b$ at the hybrid grid point $(\hat{X}_a, \hat{Y}_b, \hat{z}_k)$. All along the curve $(\hat{X}_a, \hat{z}_k)$ we have values of φ at the hybrid grid points $(\hat{X}_a, \hat{y}_j, \hat{z}_k)$. If we know the y-coordinate of each of these hybrid points, and also the y-coordinate of the new hybrid point $(\hat{X}_a, \hat{Y}_b, \hat{z}_k)$, then we can interpolate to find $\varphi(\hat{X}_a, \hat{Y}_b, \hat{z}_k)$. Again this is repeated for all the new hybrid grid points.

The final step is straightforward. Following the usual procedure, we consider the intersection of $Z = \hat{Z}_c$ with the curve $(\hat{X}_a, \hat{Y}_b)$. This is just the Lagrangian grid point $(\hat{X}_a, \hat{Y}_b, \hat{Z}_c)$. As this is a departure point we're assuming we already know its (x, y, z) coordinates. This means the interpolation can follow immediately: we know the z-coordinate of all the hybrid grid points $(\hat{X}_a, \hat{Y}_b, \hat{z}_k)$, and we know the z-coordinate of the point $(\hat{X}_a, \hat{Y}_b, \hat{Z}_c)$; so a one dimensional interpolation in the z-direction gives us $\varphi(\hat{X}_a, \hat{Y}_b, \hat{Z}_c)$.

The above procedure allows us to interpolate the data from the regular grid to all the departure points using only three one dimensional interpolations per point. It only remains to find the locations of all the hybrid grid points. This must be done before any of the interpolations can proceed.

In the cascade procedure just described, we made use of the (x, y, z) coordinates of hybrid grid points. But initially we only have the coordinates of the Lagrangian grid points. The first stage of the whole interpolation method, therefore, is to extend our

y , along the curve $(\hat{X}_a, \hat{Y}_b)$, to the point where $z = \hat{z}_k$.

Secondly we require $x(\hat{X}_a, \hat{y}_j, \hat{z}_k)$. This is slightly more complicated to achieve, and requires a two-step approach. Just as we found $y(\hat{X}_a, \hat{Y}_b, \hat{z}_k)$, we can also find $x(\hat{X}_a, \hat{Y}_b, \hat{z}_k)$. Thus we have values of x, considered as a function of y, at points along the line $(\hat{X}_a, \hat{z}_k)$. These can then be interpolated to evaluate x at $y = \hat{y}_j$. The result of this is $x(\hat{X}_a, \hat{y}_j, \hat{z}_k)$, and the algorithm is complete.

When implemented in a SL code, the method of cascade interpolation is found to be substantially faster than the Cartesian product method. The formal accuracy of the method, however, is not a simple quantity to identify. Experimentally it is found that, for reasonably smooth Lagrangian grids, the accuracy of the two methods is much the same.

Neither is grid smoothness a problem. The cascade method relies on the Lagrangian coordinates being single valued functions of the regular coordinates. This will be the case if the deformational Courant number, $\Delta t \frac{\partial u}{\partial x}$, is less than one. This condition is precisely

Chapter 6

Non-interpolating Methods

In this chapter, we shall look at an alternative form for semi-Lagrangian schemes, which avoids the need for explicit interpolation. Two problems with the interpolation stage of SL algorithms have been identified, both of which have important implications for numerical weather prediction. The first is the damping which is associated with most interpolants. This is seen to present particular problems for climate models, where model resolution is low enough to allow such damping to seriously degrade accuracy. The second problem is one of computational e□ciency: interpolation is an expensive operation, especially for three dimensional models. Ritchie [44] addresses these problems by proposing a non-interpolating version of the SL method.

Ritchie's approach is to decompose each fluid parcel trajectory into two components: one which connects grid points between time levels and a residual displacement. The scheme is designed such that the residual component is always su□ciently small to be treated stably by an Eulerian approach. The original form of his scheme has a three time-level format, but more recent work by Olim [34], and Smolarkiewicz and Pudykiewicz [53], has developed the method as a, now more conventional, two time-level scheme.

6.1 Ritchie's Scheme

Ritchie's original derivation of a non-interpolating scheme proceeds first by splitting the advecting velocity into two components. If u is the advecting velocity, then the decomposition is:

$$u = u_0 + u', \tag{6.1}$$

where u_0 is a vector field connecting grid points between time levels, and u' is a residual velocity field. Applying (6.1) to the linear advection equation

$$\frac{\partial \varphi}{\partial t} + u.\nabla\varphi = 0, \tag{6.2}$$

we obtain

$$\frac{\partial \varphi}{\partial t} + u_0.\nabla\varphi = -u'.\nabla\varphi. \tag{6.3}$$

The scheme proceeds by first assigning node-by-node values to the two velocity fields, u_0 and u'. The vector u_0 is chosen to connect the arrival node, x_i, with the node, $[x_d]$, which is closest to the departure point x_d. Once u_0 has been assigned the residual component u' is found from (6.1). For the one-dimensional problem, with a regular grid of spacing Δx, a two time-level scheme takes the following form.

For a given node, at position x_i , (see Fig. 6.1):

1) Find $[x_d]$, and let p be the number of mesh lengths from this node to the arrival point x_i.

2) Split the advection velocity: $u = p\frac{\Delta x}{\Delta t} + u'$.

3) The SL solution of (6.3) is given by:

$$\frac{\varphi(x_i, t + \Delta t) - \varphi(x_i - p\Delta x, t)}{\Delta t} = -u'\frac{\partial \varphi}{\partial x}\left(x_i - \frac{p\Delta x}{2}, t + \frac{\Delta t}{2}\right). \tag{6.4}$$

The derivative terms in (6.4) are represented by central differences.

This method can be extended, in a straightforward manner, for use in two and three dimensional problems.

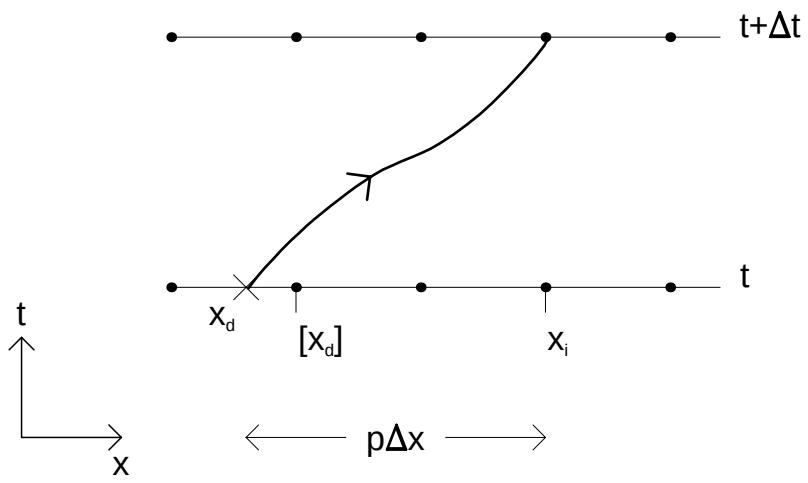


Figure 6.1: Ritchie's scheme

6.2 Olim's Scheme

Olim [34] points out that Ritchie's scheme does not totally eliminate the need for interpolation. The right hand side of (6.4) does not, in general, coincide with a grid point. Some form of interpolation must be used to carry information to the correct location. For the one dimensional scheme, this situation arises whenever p is an odd number, for then $x_i - \frac{p\Delta x}{2}$ lies between grid points. The situation becomes more complicated in two and three dimensions, for then the point of evaluation can lie at any of a considerable number of di□erent positions within each grid cell. In a truly non-interpolating scheme, the right hand side would always be evaluated at a grid point. Olim achieves this by applying the Lax-Wendro□ method, along charactntte

Oli0.971648(iy)4.86069(s)-366.594(s)4.7933(c)33.6756(h)1.9482(e)3.56312(m)2.46948(ei0.971648(iy)4

Since the field, φ, is quite general, (6.5) provides the identity

$$\left(\frac{d}{dt}\right)_r = (-\mathbf{u}'.\nabla$$

In order to obtain numerical schemes from (6.10), we must select a point, $x^{\square}$, and a contour, C.

There are three possibilities for $x^{\square}$ which lead to numerical schemes:

(i) $x^{\square} = x_i$: Eulerian schemes

(ii) $x^{\square} = x_d$: Lagrangian schemes

(iii) $x^{\square} = [x_d]$: non-interpolating SL schemes

where x_d is the departure point, at time level t^n, for the parcel trajectory which passes through $(x_i, t^n + \Delta t)$; and $[x_d]$ is the grid point nearest to x_d. Making the particular choice $x^{\square} = [x_d]$, we now look at how the contour, C, might be chosen to produce numerical schemes.

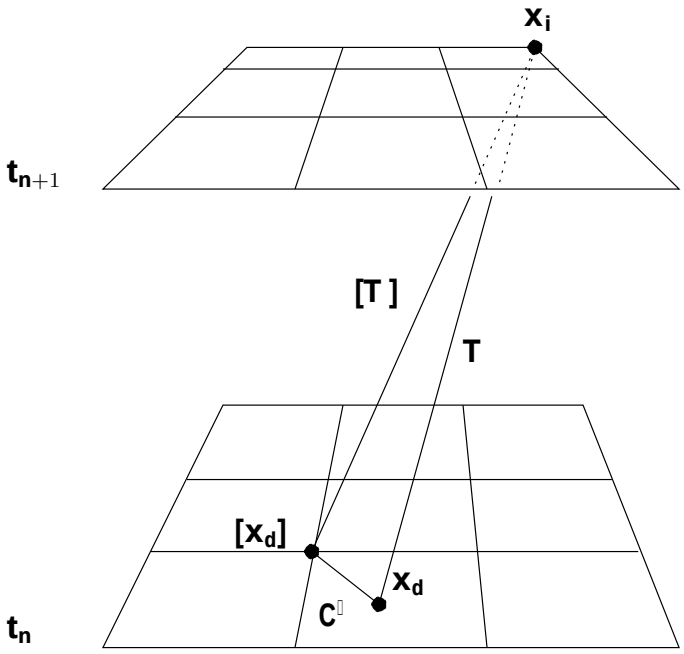


Figure 6.2: Contour Decomposition

6.3.2 Contour Decomposition

The contour C is e□ectively a route along which information held at $([x_d], t^n)$ is to be transferred to $(x_i, t^n + \Delta t)$. Smolarkiewicz and Rasch [54] investigated two classes of

contours. By introducing a little more notation, (see Fig. 6.2), we can write these as:

(A) $C = C' \cup T$, where C' is an arbitrary contour in the $t = t^n$ plane, connecting $[x_d]$ to x_d; and T is the fluid parcel trajectory from (x_d, t^n) to $(x_i, t^n + \Delta t)$.

(B) $C = [T]$, where $[T]$ is the straight line trajectory from $([x_d], t^n)$ to $(x_i, t^n + \Delta t)$.

Contour (B) leads to a class of schemes which includes Ritchie's scheme. It can be interpreted as a local coordinate transformation, which establishes a new moving frame of reference displacing grid points to grid points, over one time step. However, for reasons elaborated by Smolarkiewicz and Rasch, contour (B) poses certain design problems if we wish to derive second order time accurate schemes. Selecting contour (A), Smolarkiewicz and Pudykiewicz produce a broad class of non-interpolating schemes, which we now consider.

With the choice of contour (A), equation (6.10) becomes

$$\varphi(x_i, t^n + \Delta t) = \varphi([x_d], t^n) + \int_{C'} \nabla\varphi . dx + \int_T R\, dt. \qquad (6.11)$$

Here we have used a few simple facts about the chosen contour. For the first integral in (6.10), $dx - u dt = 0$ along the parcel trajectory T; and $dt = 0$ along C', since C' lies in the plane $t = t^n$.

We now have one last arbitrary quantity to fix, namely the little contour C'. Two particular options can be identified immediately, both of which lead to numerical schemes.

(a) C' = linear contour joining $([x_d], t^n)$ directly to (x_d, t^n), and

(b) C' = sum of linear contours parallel to the coordinate axes.

We shall first consider contour (a). If we use s to parameterize the contour, so that $s = t^n$ corresponds to the point $([x_d], t^n)$ and $s = t^n + \Delta t$ corresponds to (x_d, t^n), then (6.11) becomes

$$\varphi(x_i, t^n + \Delta t) = \varphi([x_d], t^n) - \int_{t^n}^{t^n + \Delta t} \nabla . (\varphi U)([x_d], s)\, ds + \int_T R\, dt, \qquad (6.12)$$

where $U = \frac{[x_d] - x_d}{\Delta t}$. If we ignore the source term R for the moment, we can see that (6.12) is the formal integral, between $s = t^n$ and $s = t^n + \Delta t$, of the constant coe□cient advection equation

$$\frac{\partial \varphi}{\partial s} + \nabla.(\varphi U) = 0. \qquad (6.13)$$

There are many Eulerian schemes available for the solution of such equations. In particular there are classes of schemes which possess desirable properties such as monotonicity and shape-preservation ([4], [62], [57], [12], [11]). If, together with some suitable discretisation of the source term, we apply such a scheme to (6.12) then we finally arrive at a non-interpolating SL scheme.

If we were to choose contour (b) above, then the residual equation (6.13) would be replaced with a set of one dimensional advection equations: one equation for each space dimension. Applying a one dimensional Eulerian scheme to each of these would result in a dimensional splitting scheme.

Through the above approach, we have a means of extending any Eulerian scheme into a SL scheme. This allows the CFL stability limit of Eulerian schemes to be relaxed. Through our choice of contours, (b) or (a) respectively, we can combine one dimensional schemes for use in higher space dimensions, or use fully multidimensional Eulerian schemes. The generalized scheme inherits properties of shape-preservation from the original Eulerian scheme. However, conservative Eulerian schemes do not lead to conservative SL schemes using this approach.

This approach to constructing non-interpolating schemes replaces the conventional

6.4 Non-interpolating Trajectory Methods

In Chapter 1 we described the implicit mid-point rule for solving the trajectory equation in semi-Lagrangian schemes. This method requires interpolation of the finite difference data representing the velocity field. To eliminate interpolation from all aspects of SL schemes, Smolarkiewicz and Pudykiewicz [53] apply the advection-interpolation equivalence to solving the trajectory equation. With an Eulerian advection scheme in place of a more conventional interpolation operator, such methods may be considered non-interpolating.

A more elegant route to removing computationally expensive interpolations from trajectory calculations, has been suggested by McGregor [30]. In this approach to calculating departure points, the trajectory equation is no longer solved using the implicit mid-point rule. Instead, finitj956(h)-469.8149-21eqd,ence t, and u49(a)H2.26482(y)1.9482(p)1.9482(r)-0.646948(

Taylor seres expansion of $x(t_n)$ about time $t_n + \Delta t$ may be truncated to give

$$x(t_n) \approx x(t_n + \Delta t) + \sum_{r=1}^{M} q(\Delta t)^r$$

The new velocity, $\hat{u}$, in this equation is located at the arrival point $x(t_n + \Delta t)$, which we are here assuming to be a grid point as discussed in Chapter 1. It is to be evaluated mid-way through the time step for second order time accuracy. This may be accomplished by the use of time extrapolation, again using the formulae given in Chapter 1. For the higher order derivative terms, centred di□erencing may be used. The advantages of this scheme are that it does not require complicated interpolations, and all the calculations may be carried out relatively quickly.

By calculating the product $\hat{u}.\nabla$ in spherical polar coordinates, McGregor demon-

Chapter 7

Nonlinear Advection

Introduction

Up to this point we have only considered linear advection. Referring back to Section 1.3, we see that the semi-Lagrangian method must also be applied to nonlinear advection equations. In this chapter we investigate the behaviour of SL schemes when applied to the Burgers' equation. This equation provides a simple one-dimensional analogue of the nonlinear equation for momentum, which occurs in mathematical models of fluid flow.

7.1 Burgers Equation

The viscous form of Burgers' equation is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \varepsilon\frac{\partial^2 u}{\partial x^2}, \tag{7.1}$$

where u is the flow velocity and ε a viscosity parameter. The left-hand side of this equation represents the advection of the velocity field. Advection of the field u(x, t), by the velocity u(x, t) gives rise to the nonlinear term in this equation.

We shall also be interested in the inviscid form of Burgers' equation,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0, \tag{7.2}$$

which is obtained from (7.1) by setting $\varepsilon = 0$. While solutions of the viscous Burgers' equation are always smooth, the inviscid Burgers' equation is derived from an integral conservation law which admits shocks. Kuo and Williams [19] found that the semi-

7.2 Setting up Computational Tests

Two sets of test problems will be considered, one with the viscous Burgers' equation and one with the inviscid equation. We begin by describing the experimental set-up which we shall apply to both the viscous and inviscid Burgers' equation.

For both sets of test problems the solution is computed on the domain

$$0 \leq x \leq 10.$$

Fixed boundary conditions are imposed, which are set by the initial data.

A semi-implicit semi-Lagrangian discretisation is used for the viscous Burgers' equation,

$$\frac{u^{n+1} - u_d^n}{\Delta t} = \alpha\varepsilon\delta_x^2 u^{n+1} + (1-\alpha)\varepsilon \left[\delta_x^2 u^n\right]_d, \tag{7.3}$$

where Δt is the timestep and α is the implicitness parameter. The discretisation of the inviscid equation is simply

$$u^{n+1} = u_d^n. \tag{7.4}$$

It is apparent from an examination of (7.3) and (7.4), that the SL method does not require discretisation of the nonlinear terms. The nonlinearity is subsumed within the departure point calculations and the interpolation of u to those departure points. We therefore expect to see, in the following results, some sensitivity to the choice of departure point and interpolation schemes.

Initially each test problem is solved using cubic Lagrange interpolation, for evaluating quantities at departure points. The departure points are calculated using the implicit mid-point method, with time extrapolation of the velocity and linear interpolation to trajectory mid-points. This we term the standard SL scheme. Changes to the standard scheme are then made by changing the form of interpolation and changing the departure point scheme.

7.3 Inviscid Burgers' Test Problem

Initial data for the inviscid problem is chosen to lead to the formation of a shock in the solution after a finite time. Up until this time the evolution of the solution is smooth. This provides a test of the time accuracy of the SL scheme in the early, smooth stages of the flow. At around the time of the shock formation, the design assumptions of the SL scheme no longer hold. At this point the numerical solution is liable to fail in some way.

The initial data for the inviscid problem has the form

$$u(x,0) = \begin{cases} u_{min} + (u_{max} - u_{min}) \cos^2 \left(\frac{\pi}{2} \frac{x - x_c}{a} \right), & |x - x_c| \le a \\ u_{min}, & |x - x_c| > a, \end{cases} \qquad (7.5)$$

with parameter values

$$\begin{aligned} u_{min} &= 0.0 \\ u_{max} &= 1.0 \\ x_c &= 3.0 \\ a &= 2.0. \end{aligned}$$

A g211.9552Tf5.88.9482(g)-21.8Td[(c)-6.5577]TJ/R8411.9552Tf4.21.8Td[(j)-274.284(ˇ)48/R8411.95

7.3.2 Time of Shock Formation

The method of characteristics may be used to calculate the time at which the shock first forms. Consider the inviscid Burgers' problem,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0, \quad u(x,0) = u_0(x). \tag{7.15}$$

We wish to solve this for $u(x,t)$, $t > 0$. This may be done by considering characteristics.

The characteristics of (7.15) are the solution curves of

$$\frac{dx}{dt} = u(x\text{-}0.646948]TJ/R4211.9552Tf4.560Td[(x)xx($$

we treat (7.19) as a family of curves, parametrised by x_0. The set of intersections of curves belonging to a family is termed the envelope of the family of curves. For a general family of curves, $\varphi(x, y; \lambda) = 0$, with parameter λ, the envelope is found by simultaneously solving

$$\varphi(x, y; \lambda) = 0 \tag{7.21}$$

$$\frac{\partial \varphi}{\partial \lambda} = 0. \tag{7.22}$$

with $-a \leq x_0 \leq a$ will have crossed characteristics with $x_0 > a$ at some earlier shock time. Even if this is not the case, it is still necessary to check that characteristics from these two regions of initial data do not cross earlier than time t_s. For the case $u_{min} = 0$ we find that

$$x_s = \frac{\pi + 2}{2\pi} a < a. \tag{7.29}$$

Finally, a graphical check may be made to ensure that no characteristic with x_0 in the

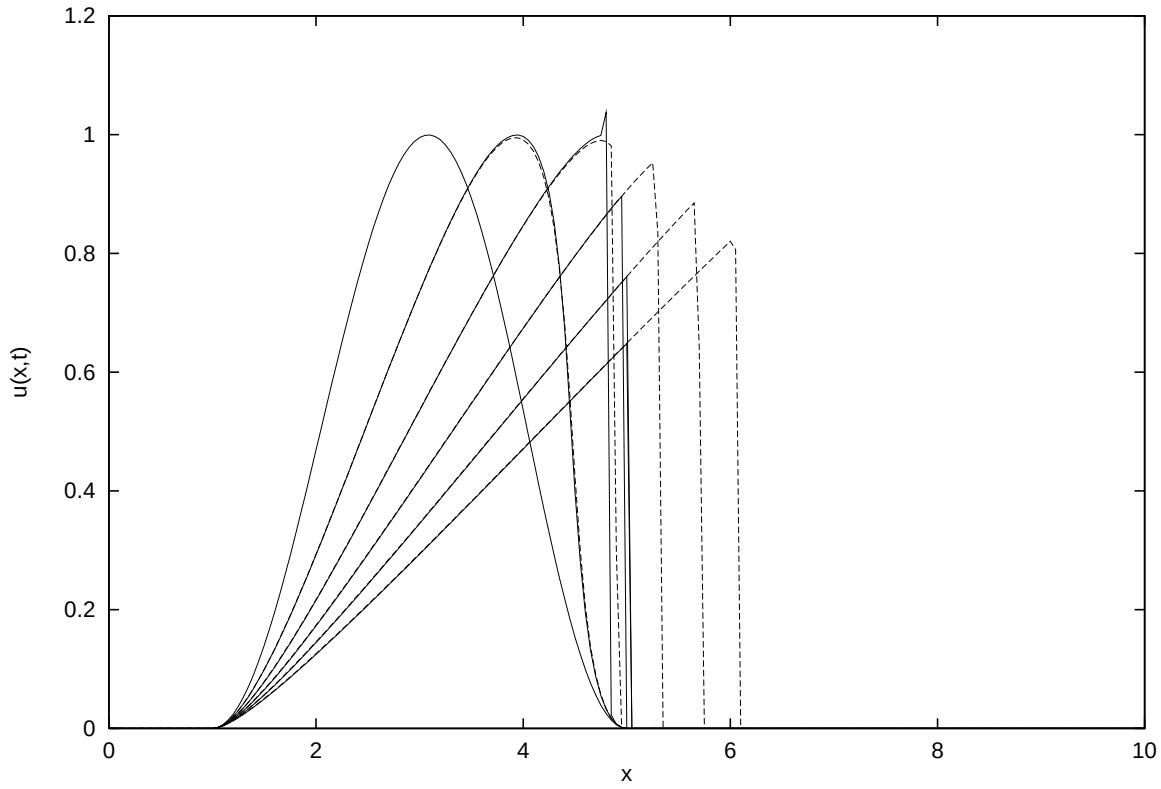


Figure 7.2: Solution of inviscid Burgers' equation using standard SL scheme: as Figure 7.1 except $u_{min} = 0$ in initial data.

the scheme's accuracy during this smooth stage of the flow.

Inviscid Test 2: Accuracy of Solution before Shock Formation.

The initial data parameters are now set as:

$$u_{min} = 0.0$$

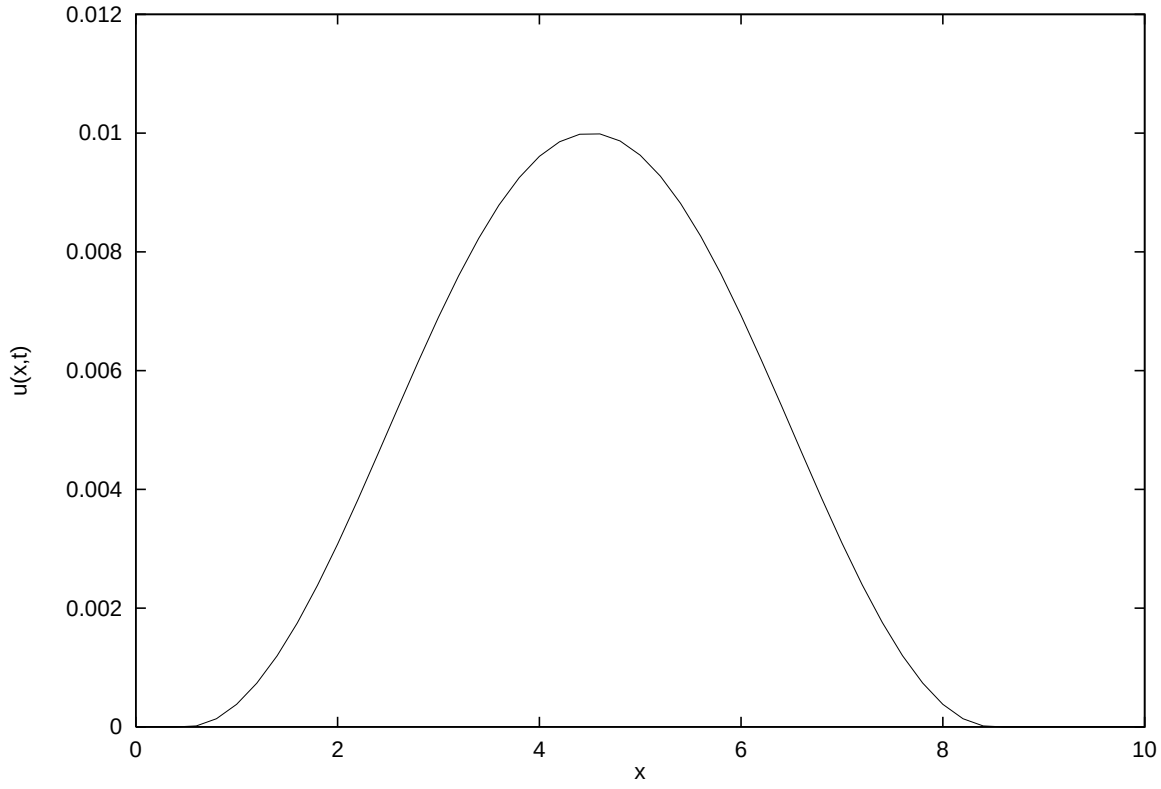
$$u_{max} = 0.01$$

$$x_c = 4.5$$

$$a = 4.0.$$

In order to pose a more severe test, both the mesh interval and the mesh ratio are doubled: $\Delta x = 0.02$, $\nu = 3.40$. For this combination of parameters the shock occurs at time $t \approx 254.65$; the timestep is $\Delta t = 0.68$. Making a total of 375 steps, and plotting the

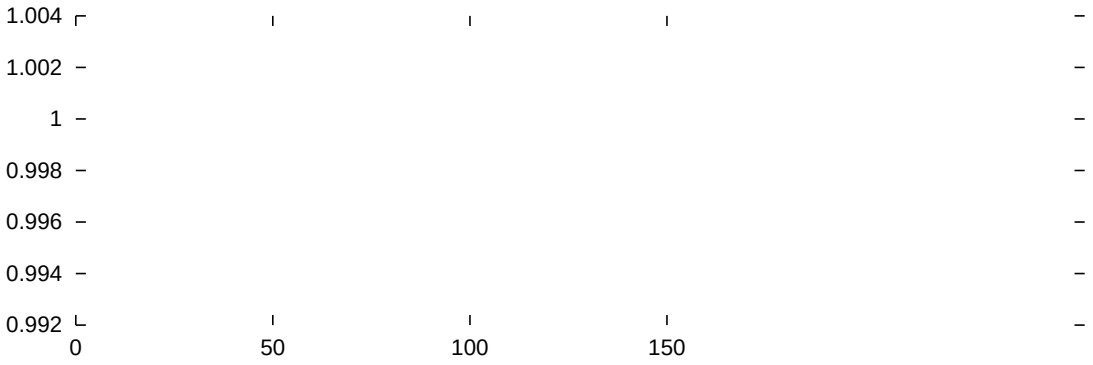
solution after every 75 steps, the solution obtained with the standard scheme is shown in Figure 7.3.

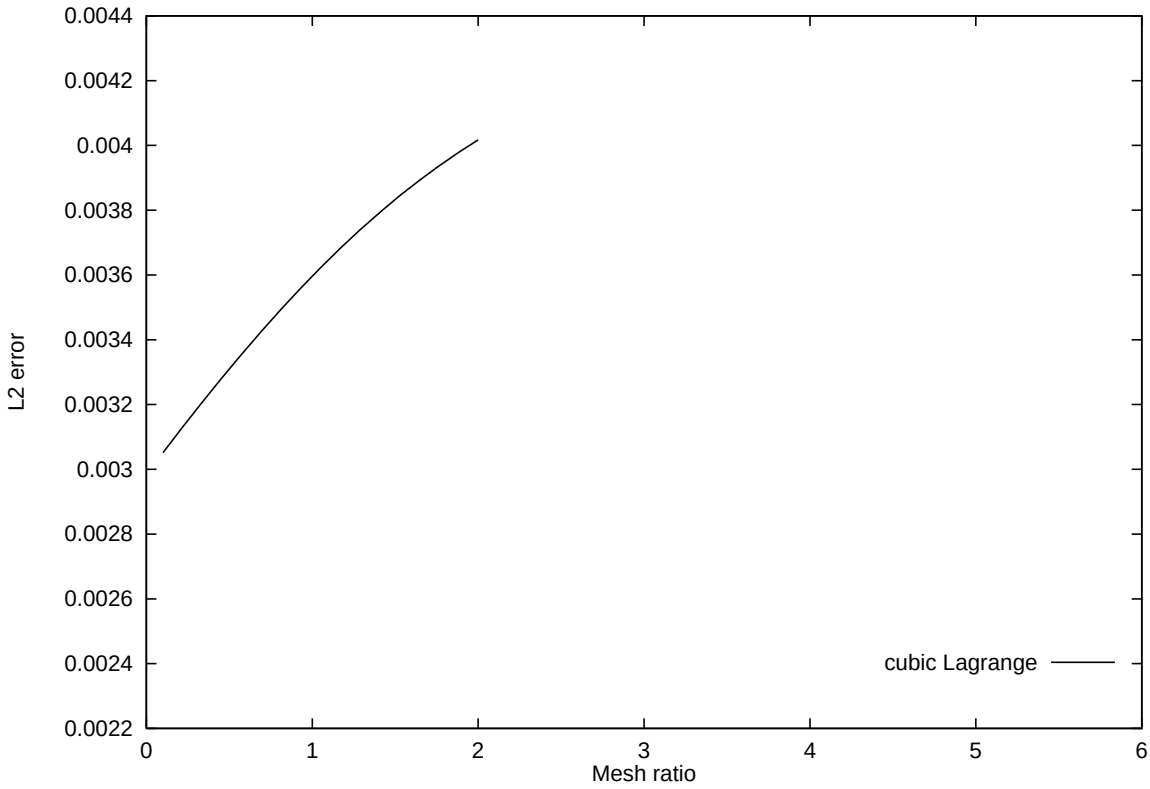


where M_n is the numerical "mass" at time t_n, given by the summation

$$M_n = \sum_{j=1}^{N} u_j^n \, \Delta x. \tag{7.34}$$

The error measurements defined by (7.32) and (7.33) allow us to compare results obtained with different SL schemes. Figure 7.4 shows error quantities plotted against


1.004
1.002
1
0.998
0.996
0.994
0.992
0
50
100
150


0.0044
0.0042
0.004
0.0038
0.0036
0.0034
0.0032
0.003
0.0028
0.0026
0.0024
0.0022
L2 error
0
1
2
3
4
5
6
Mesh ratio
cubic Lagrange

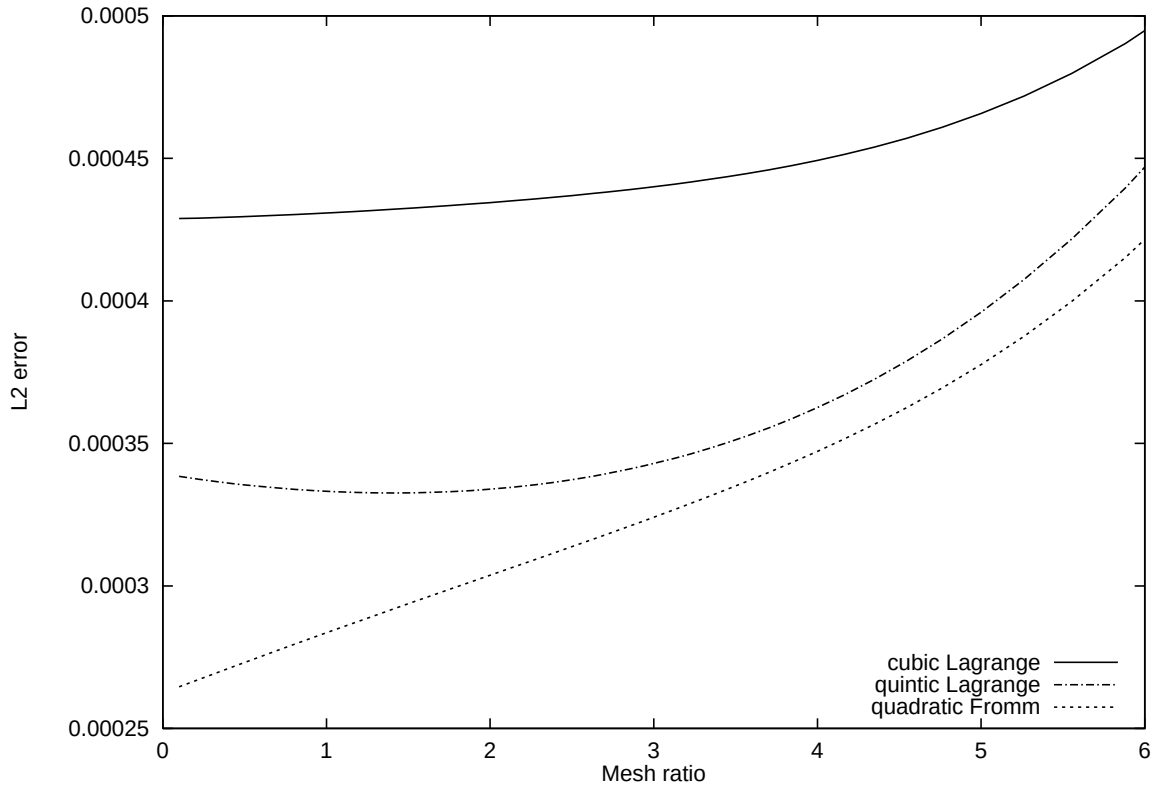


Figure 7.6: Maximum error against mesh ratio, for SL solution of inviscid Burgers' problem. The integration extends only to eighty percent of the shock time.

This is a tridiagonal system of N equations for the N unknowns, u_j^{n+1}, which we solve using the Thomas algorithm.

For the wave to move in the positive x direction we choose

$$U_L > U_R > 0.$$

7.4.2 Test 1: Departure Point Iterations

For the numerical tests the parameter values are as follows. For the initial data,

$$s = 0.6$$

$$\beta = 0.4$$

$$\varepsilon = 0.01$$

$$x_0 = 0.5,$$

and for the semi-Lagrangian scheme,

$$\Delta x = 0.025$$

$$\nu = 1.70$$

$$\alpha = 0.5.$$

The computational domain is $0 \leq x \leq 10$ and the grid has 400 equally spaced mesh intervals. With this choice of parameters there are roughly a dozen grid points within the width of the front. The integration is run for 335 timesteps. The total elapsed time is roughly 14 and the front moves through approximately 340 mesh intervals.

We begin by applying the standard semi-Lagrangian scheme, employing cubic Lagrange interpolation. Initially only two iterations are made to solve the implicit mid-point rule for determining trajectories. The result is shown in Figure 7.7. Increasing the number of departure point iterations to four, the result shows marked improvement, Figure 7.8. In view of this result we shall continue to use four iterations in calculating departure points, unless otherwise stated.

7.4.3 Error Measurements

In order to quantify the error in the numerical solution, we define two measures of error.

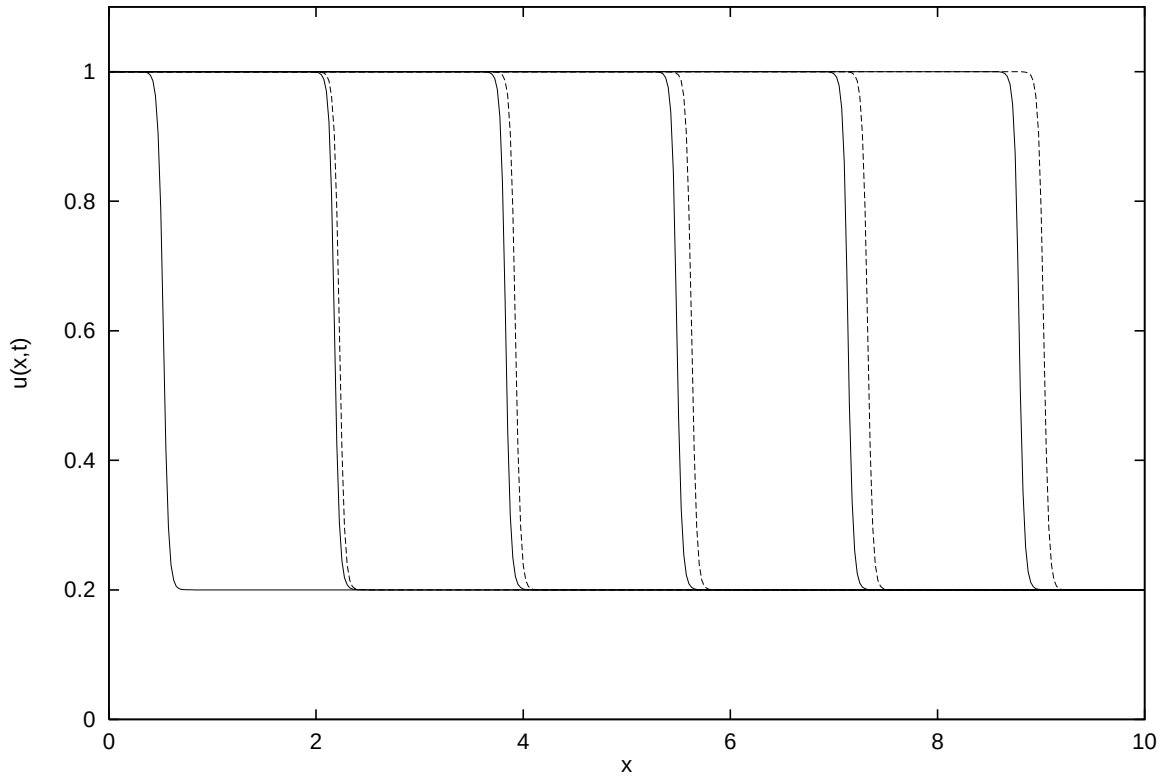


Figure 7.7: Moving front solution of viscous Burgers equation. Analytical solution (dashed lines) and SL solution (solid lines) after 0, 71, 142, 213, 284 and 355 steps. Mesh ratio $\Delta t/\Delta x = 1.7$. Two iterations for calculating departure points.

Position Error

Consider a moving front of the following form,

$$u(x,t) = \begin{cases} u_L, & x < \hat{x}_F(t) \\ u_R, & x > \hat{x}_F(t), \end{cases} \tag{7.49}$$

where $\hat{x}_F(t)$ is the position of the front at time t. At time t, let x_L be a point upstream of the front and x_R be point downstream. Integrating the solution between these points (7.49) we hhes ,

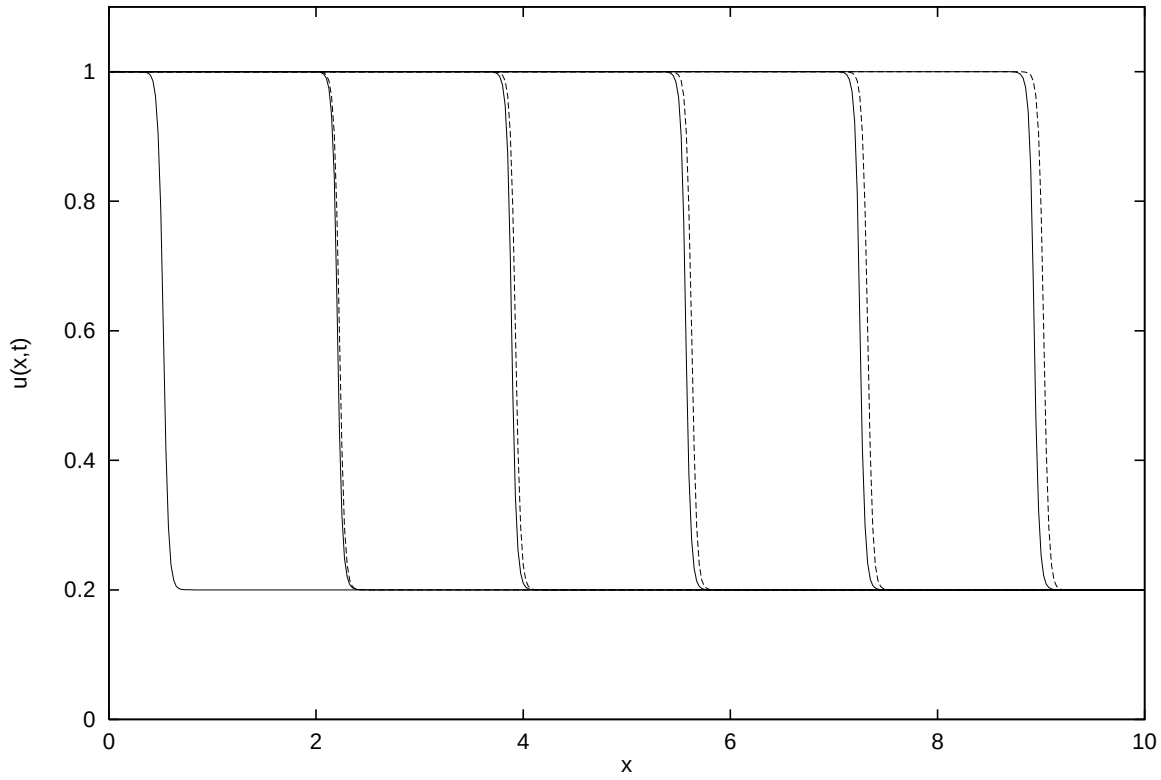


Figure 7.8: As Figure 7.7 but with 4 iterations of departure point scheme.

This may be rearranged to provide a formula for the position of the front,

$$\hat{x}_F(t) = \quad 1$$

where

$$x_L = x_0$$
$$x_R = x_{N+1}$$
$$u_L = u_0^n$$
$$u_R = u_{N+1}^n.$$

Using the numerical and analytical definitions for the position of the front at time t, we may calculate the position error at time t_n,

$$err_{pos}(t_n) = \frac{x_F(t_n) - \hat{x}_F(t_n)}{\Delta x}. \tag{7.54}$$

Shape Error

Once the position of a numerically advected front has been determined, we may calculate the L_2 error in the shape of the front. This we define by,

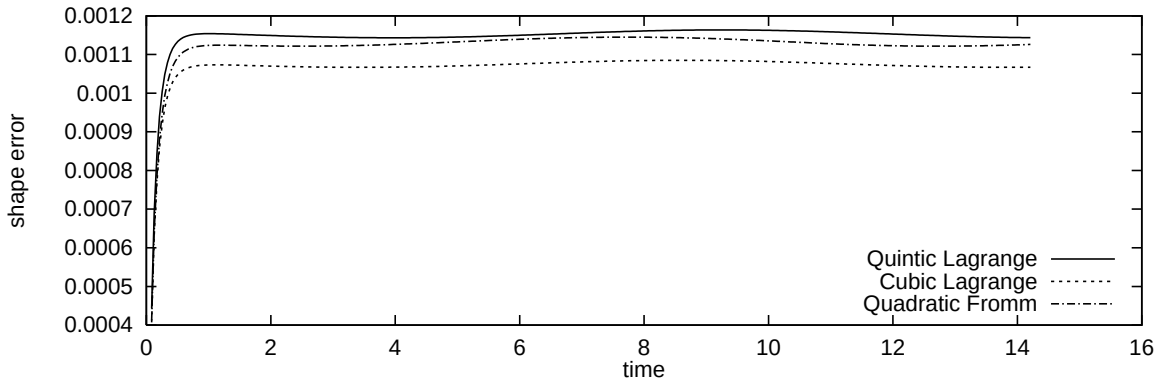
$$err_{shp}(t_n) = \left[\frac{\sum_{j=1}^{N} \left[u_j^n - u(x_j + \hat{x}_F(t_n) - x_F(t_n), t_n) \right]^2}{N} \right]^{\frac{1}{2}}. \tag{7.55}$$

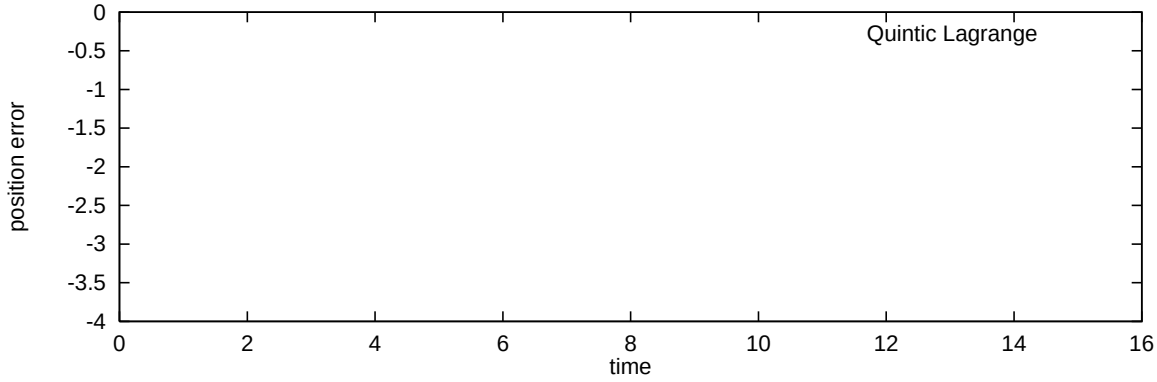
This formula shifts the analytical solution to have the same front position as the numerical solution, before calculating the difference of the two solutions. It is, therefore, a measure of the error in the shape of the front and does not reflect phase errors in the numerical solution.

7.4.4 Test 2: Interpolation

In this test we apply three different forms of interpolation: cubic and quintic Lagrange, and the centred quadratic Fromm interpolation. Again running the integration for 355 steps, the resulting errors are shown in Figure 7.9.

All three interpolants give roughly the same position error. The shape error, however, does not reflect the formal accuracy of the interpolant used. Cubic interpolation gives the most accurate results for this particular case. Quintic interpolation, which is a higher


shape error
0.0012
0.0011
0.001
0.0009
0.0008
0.0007
0.0006
0.0005
0.0004
0
2
4
6
8
10
12
14
16
time
Quintic Lagrange
Cubic Lagrange
Quadratic Fromm


position error
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-4
0
2
4
6
8
10
12
14
16
time
Quintic Lagrange

The power series expansion in t of the wind field at time $t_n + \Delta t/2$ is

$$u\left(x, t_n + \frac{\Delta t}{2}\right) = u + \frac{\Delta t}{2}\frac{\partial u}{}$$

Departure points are obtained by iterative solution of,

$$x_d = x - \Delta t u^{n+\frac{1}{2}}\left(\frac{x + x_d}{2}\right), \tag{7.66}$$

with $u^{n+\frac{1}{2}}(x)$ being evaluated by linear interpolation of the finite difference data.

Results

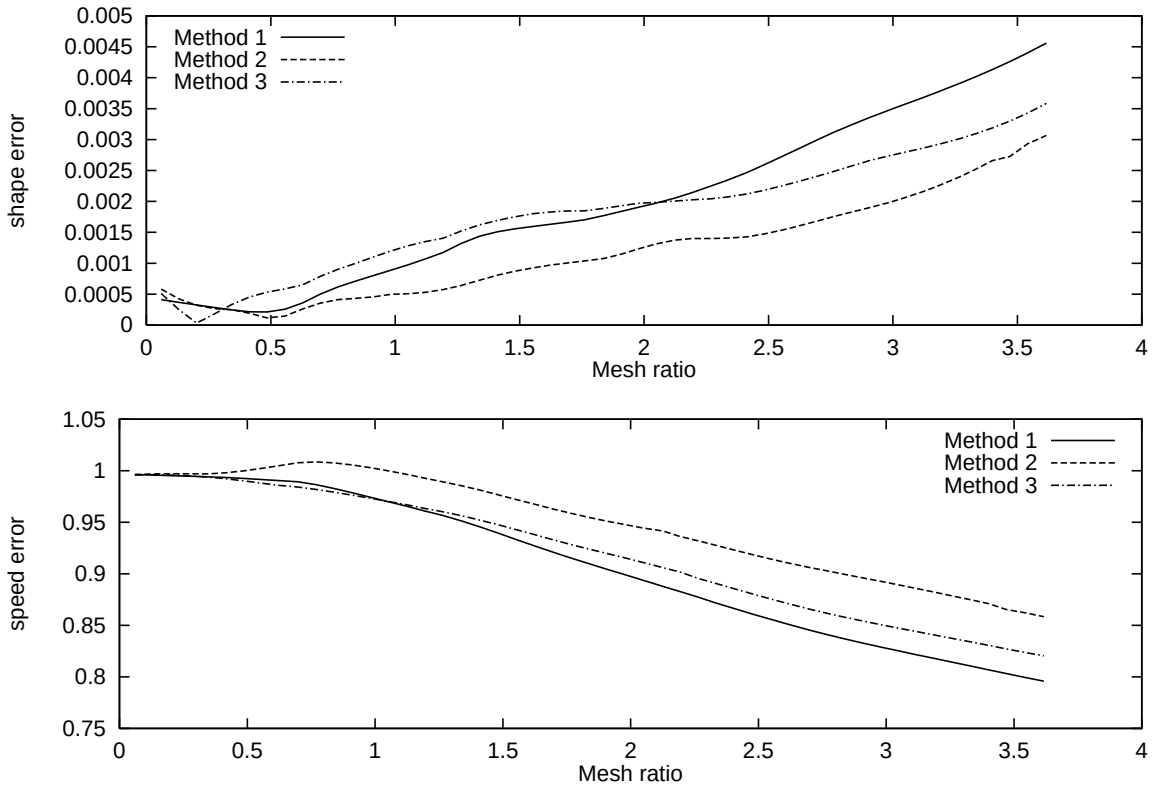


Figure 7.11: Error in speed of front and error in shape of front plotted against mesh ratio, for three di□erent methods of calculating departure points.

7.4.6 Conclusion

We conclude from this test that there may be some advantage in using methods other than the standard method for calculating departure points. This conclusion only holds, however, in the limited setting of the tests conducted here. For calculating departure points in two and three dimensions, the standard method has much in its favour, [59].

7.5 Idealised Atmospheric Flow Results

In order to show that quadratic interpolation may be advantageous in a full atmospheric model, the centred quadratic, q^M

dimensional vertical slice version of UM5.0. It also lacks the physics parametrisations of the full model, and is a stand-alone code for two-dimensional, nonhydrostatic, inviscid dynamics.

As part of the validation work for UM5.0, idealised tests have been performed with the vertical slice model, in addition to tests with shallow water and vertical column models. One such test involves a neutrally stratified atmosphere, which flows over periodic, sinusoidal terrain. By neutrally stratified, we mean that the vertical profile of temperature is such that no buoyancy forces act, when air is displaced vertically. Under conditions of stable stratification in the atmosphere, buoyancy acts as a restoring force on vertically displaced air-parcels. This mechanism is associated with one of the categories of wave motion that occur in the atmosphere, so-called gravity waves. Unstable stratification leads to unopposed ascent of air-parcels, resulting in convective upwelling. The choice of neutral stratification for our idealised test, therefore, corresponds to a test of the evanescent response of the model. Tests using stable stratification have also been made, which we do not present here.

The domain for the evanescent test has periodic lateral boundaries and free-slip boundary conditions on the vertical boundaries. The top of the model is a rigid lid, set at height $z = H$. Initially the atmosphere is in hydrostatic balance; the vertical wind field is everywhere zero, and the horizontal wind field has the same constant value, U, everywhere. The lower boundary has the sinusoidal form

$$h(x) = h_0 \cos\left[\pi \frac{2x}{L}\right]$$

$$L = 4000m.$$

The model uses a semi-implicit time integration scheme. In the following, all the implicit weights are set to one half, which corresponds to a fully time-centred Crank-Nicolson

achieve the analytical steady state, due to the dissipation inherent in the semi-Lagrangian

tests with the global three-dimensional version of UM5.0.

7.6 Conclusion

The results presented in this chapter demonstrate a potential advantage which quadratic interpolation may have over cubic, in semi-Lagrangian advection. Truncation error analysis shows cubic interpolation to have a higher formal order of accuracy than has quadratic interpolation. Consequently linear advection using a cubic semi-Lagrangian method is more accurate than with a quadratic SL scheme. This is observed unambiguously in computational results. However, in the case of nonlinear advection the correlation between truncation error and computational accuracy is less clear. Even at moderately high levels of mesh refinement, a low order interpolant may give better results than a high order one, when applied to nonlinear advection of the wind fields. This is likely to hold at the levels of grid resolution currently used in NWP models.

One explanation for this discrepancy, between analysis and computational results, is suggested by the viscous Burgers' equation test. In this test the nonlinear advection acts to increase the steepness of the front. This tendency is balanced by the diffusion term, which acts to smear out the front. In a numerical solution of this problem, the truncation error of the scheme will contribute to the smoothing effect of the diffusion. For cubic interpolation the leading coefficient has the opposite sign

semi-Lagrangian schemes be fully understood.

Chapter 8

Global Forecasting

Introduction

In the preceding chapters we have examined the application of SL methods to fluid flows in more than one dimension. Our aim is to identify SL methods particularly suitable for use in weather simulation models. Unless the model covers a very limited geographical region, it must take account of the Earth's curvature. As a first approximation we take the Earth to be a perfect sphere.

In this chapter we shall only consider modelling of atmospheric flows over the whole sphere. The special case of limited area modelling does not raise any further issues which do not arise in a global model.

The advantage of computational speed available when using a semi-Lagrangian scheme is maximised when using a regular computational grid. A regular grid allows for both e□cient interpolation and fast location of departure points. For this reason the initial development of SL methods has been to problems in domains covered by a Cartesian coordinate system, where the coordinate directions generate a regular, rectangular grid. Such methods are immediately applicable to limited area atmospheric modelling.

For extending the method for use in global simulations it is necessary to consider the problems posed by spherical geometry. These problems arise from the fact that the sphere

can not be covered by a single coordinate system without singularities. In particular, if we use equal angular spacings along lines of longitude and latitude to generate the grid, then the grid will share the singularities of the coordinate system at the poles. This singularity is reflected in the grid indexing by a degeneracy at the poles: since a line of latitude has zero length at a pole, all the grid- points along that line are coincident with the pole.

Yet for all these difficulties there are significant benefits to be obtained by the use of a latitude-longitude grid. Most importantly, finite difference calculations on such a grid have the simplicity commonly associated with a regular Cartesian grid on a plane rectangular domain. For if (λ, θ) are coordinates of longitude and latitude of points on the sphere, then in the λ-θ plane the grid is regular and rectangular. The singularity at the poles takes the form of an indexing degeneracy: along the lines $\theta = \pi/2, -\pi/2$ all points correspond to the North and South poles respectively. Unfortunately the components of velocity in this coordinate system present particular difficulties for the SL method: one of the components has a polar singularity, which must be taken into account in calculations of trajectories and momentum components near the poles.

Bates et al. [2] present perhaps the best current set of remedies for the problems of spherical geometry in SL schemes. The equations of momentum, which in component form involve polar singularities, are discretised in vector form. This avoids metric terms explicitly occurring in the discrete equations. The picture, however, is less clear for the trajectory calculation. Away from the poles the coordinate lines have low curvature, and we can apply methods of trajectory calculation designed for Euclidean geometry. Closer to the poles, a separate local coordinate system is set up for each grid point. The new coordinates are obtained by rotating the polar axis of the spherical coordinates, so that the near-polar grid point now lies on the coordinate equator. In these coordinates the trajectory equation can once again be solved using methods for flat geometry. Finally at the poles themselves a Fourier method is used. Before looking at these methods in more

8.1 Geophysical Spherical Coordinates

Let λ and θ be the longitude and latitude respectively of a point P, which is a distance r from the Earth's centre. The geophysical spherical coordinates of this point are then (λ, θ, r). It is conventional to denote the unit vectors at P in the coordinate directions λ, θ, r by $\mathbf{i}(P)$, $\mathbf{j}(P)$, $\mathbf{k}(P)$ respectively. For brevity, we shall refer to the unit vectors at any point P as simply $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$, but it is important to remember that these vectors depend on the coordinates of P.

Now let I, J, K
P

is

$$(\mathbf{v}.\nabla)\mathbf{v} = \mathbf{i}\left[\frac{u}{r\cos\theta}\frac{\partial u}{\partial\lambda} + \frac{v}{r}\frac{\partial u}{\partial\theta} + w\frac{\partial u}{\partial r} + \frac{u}{r}(w - v\tan\theta)\right] \tag{8.3}$$
$$+ \mathbf{j}\left[\frac{u}{r\cos\theta}\frac{\partial v}{\partial\lambda} + \frac{v}{r}\frac{\partial v}{\partial\theta} + w\frac{\partial v}{\partial r} + \frac{1}{r}(vw + u^2\tan\theta)\right]$$
$$+ \mathbf{k}\left[\frac{u}{r\cos\theta}\frac{\partial w}{\partial\lambda} + \frac{v}{r}\frac{\partial w}{\partial\theta} + w\frac{\partial w}{\partial r} - \frac{(u^2 + v^2)}{r}\right].$$

8.2 Shallow Water Equations on the Sphere

Many of the important aspects of global atmospheric flow are represented by the shallow water equations [60]. These result from a vertical integration of the primitive flow equations. In vector form, the shallow water equations on the sphere are:

$$\left[\frac{D\mathbf{v}}{Dt}\right]_H = -f\mathbf{r}\times\mathbf{v} - \nabla\varphi \tag{8.4}$$

$$\left[\frac{D\varphi}{Dt}\right]_H = -\varphi\nabla.\mathbf{v} \tag{8.5}$$

where φ is the geopotential height (ie. the integral of g × density through the depth of the atmosphere); f is the Coriolis parameter; r is the radius of the Earth; v is the horizontal velocity ($\mathbf{v} = u\mathbf{i} + v\mathbf{j}$) and $\left[\frac{D}{Dt}\right]_H \equiv \frac{\partial}{\partial t} + \frac{u}{r\cos\theta}\frac{\partial}{\partial\lambda} + \frac{v}{r}\frac{\partial}{\partial\theta}$. In component form, using (8.3)

$$\left[\frac{D\mathbf{v}}{Dt}\right]_H = \mathbf{i}\left[\frac{u}{r\cos\theta}\frac{\partial u}{\partial\lambda} + \frac{v}{r}\frac{\partial u}{\partial\theta} - \frac{uv}{r}\tan\theta\right] \tag{8.6}$$
$$+ \mathbf{j}\left[\frac{u}{r\cos\theta}\frac{\partial v}{\partial\lambda} + \frac{v}{r}\frac{\partial v}{\partial\theta} + \frac{u^2}{r}\tan\theta\right].$$

A discretisation of this, in the λ-θ plane, will have advective velocity components $\left[\frac{u}{\quad}\right.$

Such an approach is described by McDonald and Bates [29]. This approach is not entirely satisfactory, and in the case of the momentum equation the difficulty can be avoided by making a vector discretisation.

8.3 Vector Discretisation of Momentum Equation

Consider the nonlinear equation for horizontal advection,

$$\frac{D}$$

be expressed in terms of the basis vectors at the trajectory mid-point $\{e_\alpha(\lambda_m, \theta_m) : \alpha = 1, 2, 3\}$.

Consider the basis vectors for the spherical coordinates at the arrival point. These can be expressed relative to the Cartesian basis as follows:

$$e_\alpha(\lambda_i, \theta_i) = \sum_{\beta=1}^{3} M_{\alpha\beta}(\lambda_i, \theta_i) E_\beta \qquad \alpha = 1, 2, 3 \qquad (8.10)$$

where $M_{\alpha\beta}$ are the components of (8.2) and

8.4 Calculating Departure Points

In order to calculate the departure points used by the semi-Lagrangian method we must solve the trajectory equation. In geospherical coordinates the trajectory equation is

$$\frac{d}{dt}\begin{bmatrix}\lambda\\ \theta\end{bmatrix} = \frac{1}{a\cos\theta}\begin{bmatrix}u\\ v\cos\theta\end{bmatrix}. \tag{8.14}$$

For regions away from the poles it is possible to use the implicit mid-point rule to solve this, just as for the corresponding equation in plane geometry. Hence we make the approximation that the right hand side of (8.14) remains constant throughout a time step, and equal to its value at the trajectory mid-point. Integration of (8.14) then results in the equations

$$\begin{aligned}\lambda_d &= \lambda_i - \left[\frac{u_m^{n+1/2}}{a\cos\theta_m}\right]\Delta t\\ \theta_d &= \theta_i - \left[\frac{v_m^{n+1/2}}{a}\right]\Delta t,\end{aligned} \tag{8.15}$$

where subscript m denotes the mid-point of the great arc from (λ_d, θ_d) to (λ_i, θ_i), and the velocity components are interpolated to this point and extrapolated to the mid-time level. These equations are then solved iteratively for (λ_d, θ_d).

Sufficiently close to the poles, the assumption that the right hand side of (8.14) remains constant through a time step is no longer valid to any degree of accuracy. To deal with this problem, McDonald and Bates [29] introduce a new coordinate system for each grid point within some preset neighbourhood of a pole. Assume that node i, with coordinates (λ_i, θ_i), is such a node. Then the new coordinates, (λ', θ'), are such that node i is at $(\lambda' = 0, \theta' = 0)$ and the (i, j) unit vectors of the two coordinate systems are coincident at node i. Consideration of the geometry of this construction reveals that the

two coordinate systems are related through

$$\lambda = \lambda_i + \tan^{-1}\left[\frac{\cos\theta' \sin\lambda'}{\cos\theta' \cos\lambda' \cos\theta_i - \sin\theta' \sin\theta_i}\right]$$
$$\theta = \sin^{-1}[\cos\theta' \cos\lambda' \sin\theta_i + \sin\theta' \cos\theta_i] \qquad (8.16)$$

and that the velocity components transform according to

$$\begin{bmatrix} u' \\ v_u \end{bmatrix}$$

Before doing so, we consider what further requirements might be needed of a departure point scheme for use with the SL method. Accuracy is one issue. We shall require the method to have a second order truncation error in time. E□ciency is also a consideration, particularly in NWP applications. A departure point scheme requiring many expensive evaluations of trig functions, at each timestep, would not b

φ = **Latitude**

s = **arc length along surface of sphere.**

Time Derivatives

Let $\dot{\lambda} = \frac{d\lambda}{dt}$ and $\dot{\varphi} = \frac{d\varphi}{dt}$, then

$$\frac{d\lambda}{dt} = \frac{ds}{dt}\frac{d\lambda}{ds} \tag{8.30}$$

and

$$\frac{d^2\lambda}{dt^2} = \frac{d^2s}{dt^2}\frac{d\lambda}{ds} + \left(\frac{ds}{dt}\right)^2 \frac{d^2\lambda}{ds^2}. \tag{8.31}$$

From (8.30) and (8.31) we obtain

$$\lambda' = \frac{\dot{\lambda}}{\dot{s}} \tag{8.32}$$

$$\lambda'' = \frac{\ddot{\lambda}}{\dot{s}^2} - \frac{\ddot{s}}{\dot{s}^3}\dot{\lambda} \tag{8.33}$$

Similarly,

$$\varphi' = \frac{\dot{\varphi}}{\dot{s}} \tag{8.34}$$

$$\varphi'' = \frac{\ddot{\varphi}}{\dot{s}^2} - \frac{\ddot{s}}{\dot{s}^3}\dot{\varphi} \tag{8.35}$$

Using (8.32) and (8.33) in (8.28), and (8.34) and (8.35) in (8.29), we obtain

$$\ddot{\lambda} - 2\tan\varphi\,\dot{\lambda}\dot{\varphi} = \frac{\ddot{s}}{\dot{s}}\dot{\lambda} \tag{8.36}$$

$$\ddot{\varphi} + \cos\varphi\sin\varphi\,(\dot{\lambda})^2 = \frac{\ddot{s}}{\dot{s}}\dot{\varphi}. \tag{8.37}$$

8.5.2 Departure Point Scheme — Outline

In this section we derive a geodesic scheme for calculating departure points, which is based on using the wind fields

$$u = a\cos\varphi\,\dot{\lambda} \tag{8.38}$$

$$v = a\dot{\varphi}. \tag{8.39}$$

Trajectories are calculated using time-centred approximations, to provide second order accuracy. Given the position of a particle at time $t + \Delta t$, we wish to find its position at the earlier time t. This must be achieved numerically using only values of

wind components at the intermediate time $t + \Delta t/2$. In describing the scheme it is useful to employ the following notation:

$$\lambda^+ = \lambda(t + \Delta t), \quad \lambda^0 = \lambda(t + \Delta t/2), \quad \lambda^- = \lambda(t) \tag{8.40}$$

and similarly for other variables, so that $u^+ = u(t + \Delta t, \lambda(t + \Delta t), \varphi(t + \Delta t))$ etc.

Following the above considerations, we make Taylor series expansions about time $t + \Delta t/2$:

Longitude

$$\lambda^+ = \lambda^0 + \frac{\Delta t}{2}\dot{\lambda}^0 + \frac{\Delta t^2}{8}\ddot{\lambda}^0 + \frac{\Delta t^3}{48}\dddot{\lambda}^0 + O(\Delta t^4) \tag{8.41}$$

$$\lambda^- = \lambda^0 - \frac{\Delta t}{2}\dot{\lambda}^0 + \frac{\Delta t^2}{8}\ddot{\lambda}^0 - \frac{\Delta t^3}{48}\dddot{\lambda}^0 + O(\Delta t^4). \tag{8.42}$$

Combining (8.41) and (8.42), we replace (8.42) with

$$\lambda^- = \lambda^+ - \Delta t\dot{\lambda}^0 - \frac{\Delta t^3}{24}\dddot{\lambda}^0 + O(\Delta t^5). \tag{8.43}$$

Latitude Analogously we have

$$\varphi^+ = \varphi^0 + \frac{\Delta t}{2}\dot{\varphi}^0 + \frac{\Delta t^2}{8}\ddot{\varphi}^0 + \frac{\Delta t^3}{48}\dddot{\varphi}^0 + O(\Delta t^4) \tag{8.44}$$

$$\varphi^- = \varphi^+ - \Delta t\dot{\varphi}^0 - \frac{\Delta t^3}{24}\dddot{\varphi}^0 + O(\Delta t^5). \tag{8.45}$$

Equations (8.41) and (8.44) define λ^0 and φ^0 implicitly, in terms of the time derivatives of the coordinates at that point. The first stage of a numerical departure point scheme ohtsee

Scheme: Second Stage

Once the mid-trajectory coordinates have been found, the time derivatives of λ and φ are

Applying $\dot{s}$ = constant and $\ddot{s} = 0$ to (8.28) and (8.29), we obtain

$$\ddot{\lambda} = 2\tan\varphi\,\dot{\lambda}\dot{\varphi} \tag{8.48}$$

$$\ddot{\varphi} = -\cos\varphi\sin\varphi\,(\dot{\lambda})^2. \tag{8.49}$$

These are the constant speed geodesic equations, from which we observe that the trajectory construction of the current method does not assume $\dot{\lambda}$ and $\dot{\varphi}$ are constant during a time step. We also require the third time derivatives of λ and φ, obtained by differentiating the last two formulae:

$$\dddot{\lambda} = 2(1 + 3\tan^2\varphi)\dot{\lambda}\dot{\varphi}^2 - 2\sin^2\varphi\,\dot{\lambda}^3 \tag{8.50}$$

$$\dddot{\varphi} = -$$

8.5.5 E□cient use of Trig Calculations

To solve (8.54) and (8.55) iteratively would require repeated calculations of $\tan\varphi^0$ and $\sec\varphi^0$ for each iteration, at every grid point, at every time step. A considerable gain in e□ciency is made by calculating trig functions just once and storing their values for every grid point. We therefore wish to replace the trig functions in (8.54) and (8.55) with functions of φ^+, rather than of φ^0. Our approach is to adapt the procedure described in [45].

We begin by considering

$$\varphi^+ = \varphi^0 + \frac{\Delta t}{2}\dot{\varphi}^0 + \frac{\Delta t^2}{8}\ddot{\varphi}^0 + \frac{\Delta t^3}{48}\dddot{\varphi}^0 + O(\Delta t^4).$$

For a constant speed trajectory along a great circle we may make use of (8.55) to obtain

$$\varphi^+ = \varphi^0 + \frac{\Delta t}{2}\frac{v^0}{a} - \frac{\Delta t^2}{8}\frac{(u^0)^2}{a^2}\tan\varphi^0 - \frac{\Delta t^3}{48}\frac{(u^0)^2 v^0}{a^3}(3\sec^2\varphi^0 - 2) + O(\Delta t^4). \qquad (8.56)$$

The quantity we require, φ^0, appears implicitly in this equation. To obtain a solution we assume φ^0 has an expansion of the form

$$\varphi^0 = \varphi^+ + p_1\Delta t + p_2\Delta t^2 + p_3\Delta t^3 + O(\Delta t^4). \qquad (8.57)$$

$)^2$ Substituting (8.57) into (8.56),

Hence

$$\varphi^0 = \varphi^+ - \frac{v^0}{2a}\Delta t + \frac{(u^0)^2}{8a^2}\tan\varphi^+\Delta t^2 - \frac{(u^0)^2 v^0}{24a^3}\Delta t^3 + O(\Delta t^4). \qquad (8.62)$$

This is now used to evaluate the various trig functions required in the departure point scheme. The Taylor expansions of tan and sec for a small displacement δ about the point x are

$$\tan(x+\delta) = \tan x + \delta\sec^2 x + \delta^2\sec^2 x\tan x + O(\delta^3) \qquad (8.63)$$

$$\sec(x+\delta) = \sec x + \delta\sec x\tan x + \frac{\delta^2}{2}\sec x(\sec^2 x + \tan^2 x) + O(\delta^3). \qquad (8.64)$$

These expansions give

$$\tan\varphi^{[()\overset{+}{0}\overset{v}{6414}(a)226309n04[(+)169456]411955293()356312(x)44]42624922]116}$$

This departure point scheme is identical to the Ritchie and Beaudoin scheme, when applied in the context of a two time-level SL advection scheme. The procedure which we have followed here, however, has the potential to be applied in curved spaces other than the surface of the sphere. Possible applications for the method may be found, for instance,

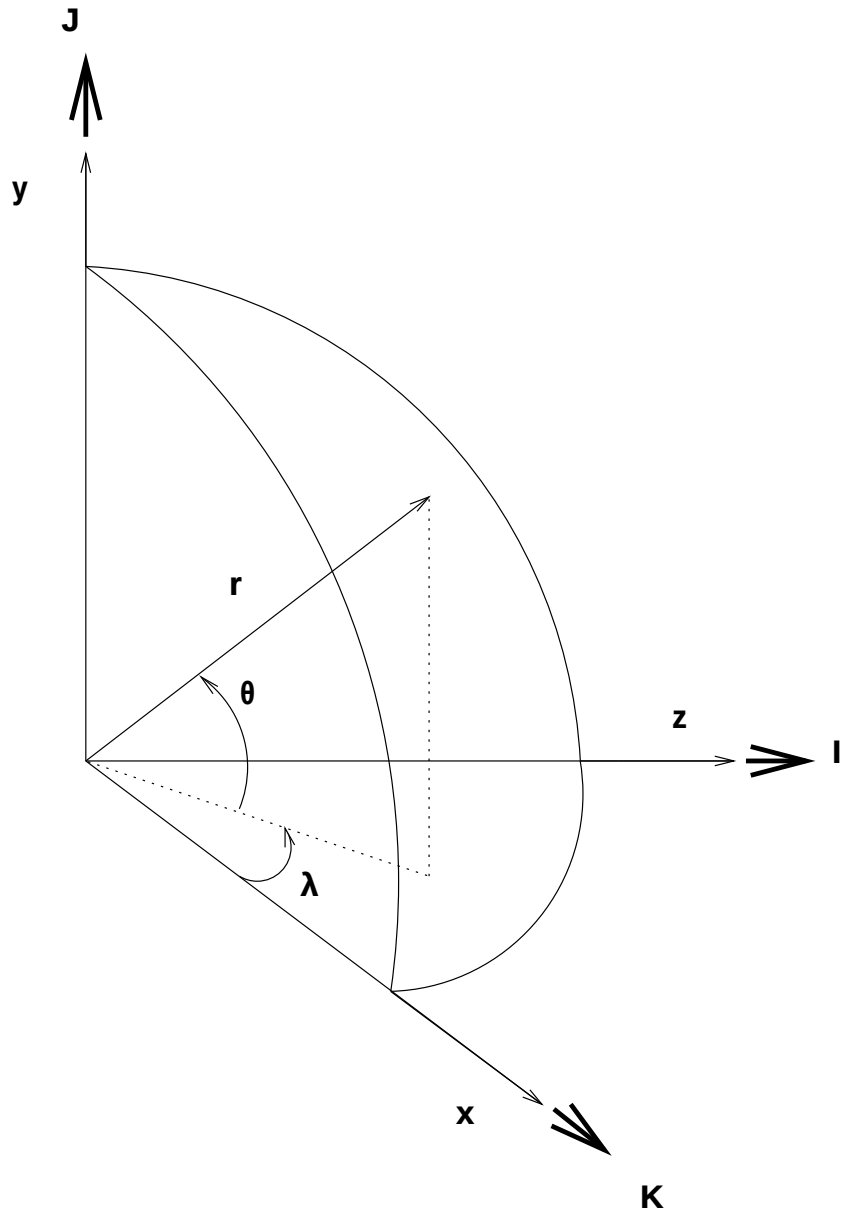


Figure 8.1: Geophysical spherical coordinates

Chapter 9

Mass Conserving Schemes

Before considering conservation of mass in numerical schemes it is first helpful to consider what we require of a numerical simulation. An issue of central importance in any simulation of fluid flow is the degree to which the simulation exhibits the true physics of the flow. Many of the physical principles we expect to be obeyed by a fluid, though not all, can be expressed as the conservation of some quantity. For instance, in the regime of classical physics, we would expect an isolated fluid system to have constant values of total mass, momentum and energy.

In cons2(r)-0.64694d um t aing cr24.79152(a)-2.26432(s)4.7648174(i)0.97164em

spurious physical processes when representing atmospheric dynamics. Such errors could obscure the essential interactions to be represented by the climate model. In practice, however, this ideal can never be reached. But the manner in which we approximate to the ideal will have a strong qualitative effect on the end result.

A given property of a mathematical model, such as a conservation principle, may translate across to the numerical model in one of two ways. It may be the case that a direct numerical analogue of the original property is found to hold for the discrete equations, at all levels of mesh refinement. Alternatively it could be the case that no such property holds, except in the asymptotic limit of refinement. In weather prediction short term accuracy is the desired goal. Under such a regime it may be sufficient to approach certain properties of the flow only in the asymptotic limit, provided that this allows for an advantage in computational speed. The situation is entirely different for climate prediction, where a strong case can be made for the numerical model to possess as many exact analogue properties as possible.

We are now in a position to ask how does the semi-Lagrangian method fare in these respects. At present the answer to this question is still largely incomplete. Until recently no SL method existed which preserved total advected mass. Analysis of deeper dynamical properties is wholly lacking.

For the remainder of this chapter, we shall turn our attention specifically to conservation of mass, and other advected quantities, in numerical schemes.

9.1 Conservation and Continuity

Conservation is a global property. For an isolated fluid system we expect a global quantity, such as total mass, to be conserved. The total mass after any period of time should be precisely the same as it was initially.

If the system is not isolated then quantities which are free to be exchanged with the fluid's surroundings will not be conserved within the system itself. For instance, to a

good level of approximation the atmosphere does not lose or gain mass on a short time

Secondly, internal processes are modelled by the generation of m through a source term s defined at all points in V ,

$$\frac{d_i}{dt}M = \int_V s\, dx. \qquad (9.4)$$

is being carried along within the fluid. For instance this could be a gas being released into the air-stream from an industrial outlet. As a further restriction we assume that this substance is merely passively advected, without any other processes a□ecting its motion. Towards the end of the chapter we shall examine the complications which arise in the case of transport of momentum, in which case nonlinearities appearing in (9.6) need to be dealt with appropriately.

Consider a fluid of density $\rho(x,t)$ which occupies a region partitioned into cells $\{C_j\}$ with corresponding volumes $\{V_j\}$. Define the cell-average value of ρ,

$$\bar{\rho}_j(t) = \frac{1}{V_j}$$

handle these volume derivatives as source terms. A scheme involving more conventional flow variables can be obtained by using the equivalency of volume change and velocity divergence,

upstream departure region of cell j are labelled subscript d(j). The scheme (9.20) now takes the form

$$\rho_j^{n+1} V_j = \rho_{d(j)}^n V_{d(j)}^n. \tag{9.21}$$

We shall use $V_{d(j)}^n$ to refer to both the departure region and its volume, allowing context to dictate the intended meaning.

Just as in the standard semi-Lagrangian framework, the data required by this scheme at the departure time level (t_n) are not immediately available. From the representation of ρ on the Eulerian grid, $\{\rho_j^n, V_j\}$, a Lagrangian representation $\{\rho_{d(j)}^n, V_{d(j)}^n\}$ must be

If the total volume of fluid to be modelled remains constant, then this ought to be built into the scheme as a constraint :

$$\sum_j V_j = \sum_j V^n_{d(j)} \text{ for all } n.$$

However, we may go beyond this global constraint and identify a natural local constraint. From the theory of differential equations we know that fluid particle trajectories do not intersect one another. This can be represented numerically by requiring the approximations of the regions $V^n_{d(j)}$ to form a partition of the whole domain. That is, they should cover the domain without gaps or overlaps. The result is a collection of control-volumes whose structure may be associated with a (usually irregular) grid. For instance, such a grid is formed by the edges of control-volumes. This grid approximates to the true Lagrangian grid, formed between the regions which flow into separate Eulerian cells after one time-step, figure (9.1).

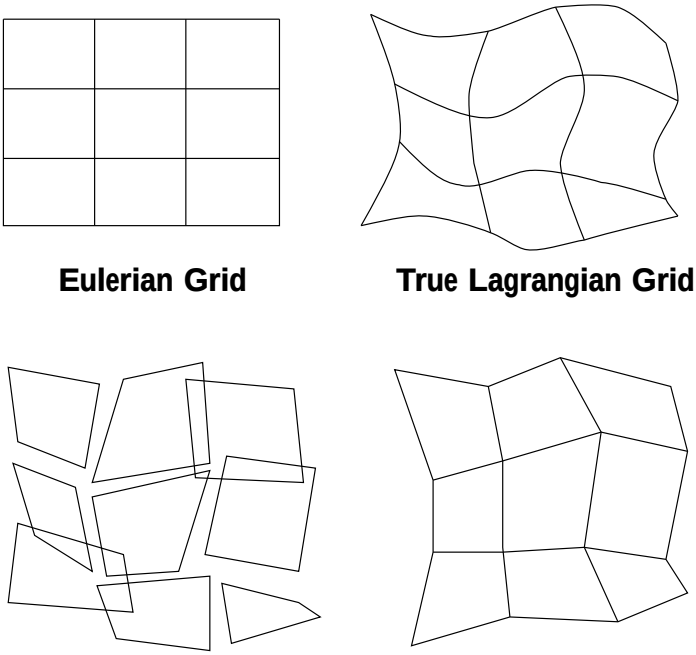


Figure 9.1: Control Volumes and Grids

The process of numerically transferring a discrete representation of a function from one grid to another, both occupying the same domain, is known as rezoning. The rezoning

constraint, described above, allows the right-hand side of (9.21) to be split in two parts for e□cient numerical evaluation. The splitting results from identifying the regions of intersection between the Lagrangian and Eulerian control-volumes/grid cells. We shall next describe how this allows the right-hand side of (9.21) to be discretised in such a way that (9.23) is satisfied.

Since the whole rezoning process involves data only at the departure time level t_n, we shall drop the n superscript for the remainder of this description.

In general $V_{d(j)}$ will overlap with more than one grid cell, perhaps containing some entirely. Let C_{ij} be the region of intersection between V

speed this is equivalent to using cubic interpolation in a standard semi-Lagrangian scheme, Plante [35].

A practical rezoning strategy, to obtain the departure regions $V_{d(j)}$, is given by connecting departure points of cell vertices with straight edges. This produces departure zones which are general quadrilaterals in two dimensions. With this geometry analytic evaluation of the integrals appearing in the second term of (9.24) is possible, though cumbersome. This suggests that greater sophistication in the representation of departure zones would be computationally inefficient.

In three dimensions the corresponding departure regions are three-dimensional figures with bilinear faces. Identifying regions of intersection with Eulerian grid cells — and performing exact integration — in this geometry would appear to be a serious challenge. Further geometric simplifications are likely to be needed.

9.4.2 Downstream Scheme

It is possible to invert the procedure outlined for the upstream scheme, and obtain a downstream discretisation for the $\overline{\rho}$ conservation equation (9.19). And again this scheme takes the basic form (9.20). In such a scheme the departure region, V_d, is taken to be a grid-cell. From this beginning, the aim is to construct a numerical procedure for distributing the mass contained in this cell to grid-cells at the arrival time. Just as before, the transfer of mass between time levels depends on how we model local volume changes. But in the place of interpolation of the field variable we now have redistribution between grid-cells. The numerical scheme takes the form

$$\rho_j^{n+1} V_j = \sum_j m_{ij}^n, \tag{9.26}$$

where m_{ij} is the mass of fluid which flows from cell i, at time t_n, to cell j, at time t_{n+1}.

The mass distribution, m_{ij}, is calculated in two stages. First the region within cell i which flows to cell j must be identified. Secondly the mass within this region is calculated

where

$$\beta_j = (u_{Hj}^{n+1} - u_{Lj}^{n+1})V_j.$$

The task is now to find the largest values of α_j, in the range $0 \le \alpha_j \le \alpha_j^{\max}$, which satisfy the conservation condition (9.30). It may be assumed that the QMSL algorithm produces correction terms with a total mass larger than the discrepancy δM. If these two masses were the same, then the QMSL algorithm would, fortuitously, be conservative. In such a case further adjustment to the solution would be unnecessary. Alternatively, if the QMSL corrections have a mass too low to meet the conservation requirement, then a simple change of sign reverses this ordering :

$$\beta_j = (u_{Lj}^{n+1} - u_{Hj}^{n+1})V_j$$

$$\delta M = M_L - M_0.$$

4:

Chapter 10

Numerical Experiments with Conservative Schemes

10.1 Introduction

In this chapter we describe a two-dimensional scheme of the form outlined in the previous chapter. This is a downstream scheme, which identifies overlap regions between the Eulerian and Lagrangian grids. These regions, which are polygons, are broken down into triangular regions. Polynomial recovery is applied to the finite difference data on the Eulerian grid. The polynomials are then integrated over each triangle to transfer the representation of the data to the Lagrangian grid. Rather than using some approximation technique for the integration, we shall integrate the polynomials exactly. The resulting scheme conserves mass to machine accuracy.

A comparison is made between this scheme and the quasi-conservative scheme of Priestley, described in Chapter 5. Both these schemes are compared against two non-conservative schemes. The first of these is a basic semi-Lagrangian scheme without any control for monotonicity. The second scheme is the same as the first, but with the addition of the quasi-monotone method of Bermejo and Staniforth, described in Chapter 2. All of the schemes use Lagrange polynomials for interpolation, and as basis functions for the

remapping scheme. The same method of trajectory calculation is used for all the schemes.

10.2 Remapping Scheme

A remapping scheme has two requirements :

- a method for calculating overlap regions of Eulerian and Lagrangian grids
- integration of a representation of the solution in the overlap regions.

Schemes other than the semi-Lagrangian schemes described here employ algorithms for these or similar functions. For instance the Lagrange-Galerkin method [18], [17].

10.2.1 Exact Integration

The integration method to be described here is based on a technique for exact integration of polynomials over arbitrary triangles. An extension of this method to three-dimensional geometry is possible.

As described in the introduction to this chapter, the overlap regions between the two grids are analyzed into triangles. A polynomial recovery of the finite difference data must be integrated over each triangle. In this section we describe a technique for the exact integration of a bi-variate polynomial over an arbitrary triangle. This technique consists of five stages :

[1] Express the polynomial as a product of linear factors

[2] Find the transformation for mapping the unit right-angled triangle onto the given triangle

[3] Apply the transformation found in [2] to each linear factor of the polynomial

[4] Collect terms of the same power within the polynomial

[5] Integrate over unit triangle.

In stage [1

Step [3] requires the transformation to be applied to each linear factor of the polynomial, in order to find the polynomial in the transformed coordinates. Let $\tilde{p}(\xi,\eta)$ be the transformed polynomial. From (10.1) and (10.2) we obtain

$$\begin{aligned}\tilde{p}_n(\xi,\eta) &= \prod_{r=1}^{n} [a_r(J_{xx}\xi + J_{xy}\eta) + b_r(J_{yx}\xi + J_{yy}\eta) + J_x a_r + J_y b_r + c_r] \\ &\equiv \prod_{r=1}^{n} (A_r\xi + B_r\eta + C_r) \qquad (10.3)\end{aligned}$$

where

$$A_r = J_{xx}a_r + J_{yx}b_r$$

$$B_r = J_{xy}a_r + J_{yy}b_r$$

$$C_r = J_x a_r + J_y b_r + c_r.$$

The final step will involve integrating this transformed polynomial over the unit triangle. This task is considerably simplified by step [4], in which terms of the same power are collected together. So, we wish to find the coefficients Q_{rs} for which

$$\tilde{p}_n(\xi,\eta) = \sum_{r=0}^{n}\sum_{s=0}^{k-r} Q_{rs}\xi^r\eta^s$$

A recurrence procedure offers a simple means of calculating these coefficients.

The recurrence builds through polynomials of increasing order $k = 1,\ldots,n$, by successively multiplying by linear factors, until the polynomial (10.3) is reached. The first level of the recurrence has coefficients $Q_{rs}^{[0]}$, where $r = 0,1$ and $s = 0,r$. These coefficients define the first-level polynomial, which is simply equal to the first linear factor of (10.3) :

$$Q_{10}^{[0]}\xi + Q_{01}^{[0]}\eta + Q_{00}^{[0]} \equiv A_1\xi + B_1\eta + C_1,$$

from which we obtain

$$Q_{00}^{[0]} = C_1$$

$$Q_{01}^{[0]} = B_1$$

$$Q_{10}^{[0]} = A_1.$$

Next, assume the recurrence has been completed up to level k, and we wish to add level k + 1. If the level k polynomial is

$$\tilde{p}_k(\xi,\eta) = \prod_{r=1}^{k}(A_r\xi + B_r\eta + C_r) \equiv \sum_{r=0}^{k}\sum_{s=0}^{k-r} Q_{rs}^{[k]}\xi^r\eta^s,$$

then the polynomial at the next level of the recurrence is

$$\begin{aligned}\tilde{p}_{k+1}(\xi,\eta) &= (A_{k+1}\xi + B_{k+1}\eta + C_{k+1})\tilde{p}_k(\xi,\eta)\\ &= (A_{k+1}\xi + B_{k+1}\eta + C_{k+1})\sum_{r=0}^{k}\sum_{s=0}^{k-r} Q_{rs}^{[k]}\xi^r\eta^s\\ &\equiv \sum_{r=0}^{k+1}\sum_{s=0}^{r} Q_{rs}^{[k+1]}\xi^r\eta^s.\end{aligned}$$

To find the new coefficients $Q_{rs}^{[k+1]}$, we work with $\tilde{p}_k$ written in the form of a univariate polynomial in ξ :

$$\tilde{p}_k(\xi,\eta) = \sum_{r=0}^{k} q_r^{[k]}(\eta)\xi^r,$$

where the coefficients are polynomials in η,

$$q_r^{[k]}(\eta) = \sum_{s=0}^{k-r} Q_{rs}^{[k]}\eta^s. \qquad (10.4)$$

Using this form we calculate $\tilde{p}_{k+1}$:

$$\begin{aligned}\tilde{p}_{k+1}(\xi,\eta) &= (A_{k+1}\xi + B_{k+1}\eta + C_{k+1})\sum_{r=0}^{k} q_r^{[k]}(\eta)\xi^r\\ &= A_{k+1}q_k^{[k]}(\eta)\xi^{k+1} + \sum_{r=1}^{k}\left(A_{k+1}q_{r-1}^{[k]}(\eta) + B_{k+1}\eta q_r^{[k]}(\eta)\right)\xi^r +\\ &\quad B_{k+1}\eta q_0^{[k]}(\eta) + C_{k+1}\sum_{r=0}^{k} q_r^{[k]}(\eta)\xi^r,\end{aligned}$$

from which we obtain recurrence relations for the coefficients of ξ^r :

$$\begin{aligned}q_{k+1}^{[k+1]}(\eta) &= A_{k+1}q_k^{[k]}(\eta)\\ q_r^{[k+1]}(\eta) &= A_{k+1}q_{r-1}^{[k]}(\eta) + B_{k+1}\eta q_r^{[k]}(\eta) + C_{k+1}q_r^{[k]}(\eta) \quad r = 1,\ldots,k\\ q_0^{[k+1]}(\eta) &= B_{k+1}\eta q_0^{[k]}(\eta) + C_{k+1}q_0^{[k]}(\eta).\end{aligned}$$

Applying the definition (10.4) of the $\tilde{q}_k(\eta)$ polynomials in terms of the coefficients Q_{rs}, we obtain recurrence relationships for the coefficients :

$$Q^{[k+1]}_{k+1\,0} = A_{k+1} Q^{[k]}_{k\,0}$$

$$Q^{[k+1]}_{r\,0} = A_{k+1} Q^{[k]}_{r-1\,0} + C_{k+1} Q^{[k]}_{r\,0} \qquad r = 1, \ldots, k$$

$$Q^{[k+1]}_{rs} = A_{k+1} Q^{[k]}_{r-1\,s} + B_{k+1} Q^{[k]}_{r\,s-1} + C_{k+1} Q^{[k]}_{rs} \qquad r = 1, \ldots, k;\ s = 1, \ldots, k-r$$

$$Q^{[k+1]}_{r\,k+1-r} = A_{k+1} Q^{[k]}_{r-1\,k+1-r} + B_{k+1} Q^{[k]}_{r\,k-r} \qquad r = 1, \ldots, k$$

$$Q^{[k+1]}_{0\,k+1} = B_{k+1} Q^{[k]}_{0\,k}$$

$$Q^{[k+1]}_{0\,s} = B_{k+1} Q^{[k]}_{0\,s-1} + C_{k+1} Q^{[k]}_{0\,s}$$

$$Q^{[k+1]}_{0\,0} = C_{k+1} Q^{[k]}_{0\,0}.$$

Once the coefficients Q_{rs} have been found, stage [5] of the algorithm is to perform the integration,

$$I = \int_{0}^{1} \int_{0}^{1-\xi} \sum_{r=0}^{n} \sum_{s=0}^{n-r} Q_{rs} \xi^r \eta^s \, d\eta \, d\xi.$$

Simpson's rule for double integrals [illegible] (94-377.8

where the canonical space now has coordinates (

for its severe lack of conservation.

Chapter 11

Spurious Orographic Resonance

behaviour occurs fall well within the operational range of weather simulation.

The solution proposed in Rivest et al. is to o□-centre the weighting between explicit and implicit parts, in the semi-implicit discretisation. For an equation of the form

$$\frac{DF}{Dt} = G, \tag{11.1}$$

a one parameter family of discretisations is considered :

$$\frac{DF}{Dt} \rightarrow \frac{1}{\Delta t}\left[F^n - F^{n-1}\right] \tag{11.2}$$

$$G \rightarrow \frac{1+\delta}{2} G^n + \frac{1-2\delta}{2} G^{n-1} + \frac{\delta}{2} G^{n-2} \tag{11.3}$$

where $H^{n-r} \equiv H(x(t_n - r\Delta t), t_n - r\Delta t)$. This form of discretisation retains the $O(\Delta t^2)$ accuracy of the centred approximation ($\delta = 0$) by introducing a third time-level. At this new time-level, departure points must be found and interpolations performed there, just as for the $t - \Delta t$ time-level.

Repeating the Fourier analysis of orographically forced standing waves, it is found that for this new discretisation resonance will not occur whenever δ is non-zero. Values of δ away from zero actively inhibit the formation of persistent standing wave patterns. Stability analysis for the new discretisation shows that a necessary condition for stability is

$$\delta \geq 0.$$

The choice $\delta = 1/2$ is recommended by Rivest et al., since this produces some simplification when applied to (11.3) while eliminating orographic resonance.

Before considering further developments of this scheme, we shall outline the various options for calculating departure points at the extra time-level introduced in (11.3).

Two methods for calculating departure points at the new time-level are examined by Rivest et al. In the first method it is assumed that a departure point at time $t - \Delta t$ has already been found using standard methods. The trajectory thus found is extrapolated, along a great-circle arc, back to the $t - 2\Delta t$ time-level to give a departure point there.

The second method requires considerable extra computational time compared to the first. In this approach the extra level of departure points is found using the same

techniques as for the first level. For instance if the departure points at time $t - \Delta t$ are calculated using the implicit mid-point rule,

$$x(t - \Delta t) = x(t) - \Delta t\, u\left[\frac{1}{2}[x(t) + x(t - \Delta t)], t - \frac{1}{2}\Delta t\right], \qquad (11.4)$$

then the same rule is applied to calculating the departure points at time $t - 2\Delta t$. In (11.4) the velocity field at the intermediate time-level is not immediately available, but must be obtained by extrapolation of the field at time-levels $t - \Delta t$ and $t - 2\Delta t$.

For calculating departure points at $t - 2\Delta t$ there are two ways in which the implicit mid-point rule might be applied. In one, a piecewise trajectory between times t and $t - 2\Delta t$ is constructed. In the other, two separate trajectories are calculated.

Considering the piecewise method first, the part of the trajectory between t and $t - \Delta t$ is simply that found by solving (11.4). The part between $t - \Delta t$ and $t - 2\Delta t$ is obtained from

$$x(t - 2\Delta t) = x(t - \Delta t) - \Delta t\, u\left[\frac{1}{2}[(x(t - \Delta t) + x(t - 2\Delta t)], t - \frac{3}{2}\Delta t\right]. \qquad (11.5)$$

This formula too involves evaluation of velocities at an intermediate time-level. Rather than extrapolating from previous time-levels, interpolation may be used here between data at times $t - \Delta t$ and $t - 2\Delta t$.

However, an alternative formulation of the implicit mid-point rule avoids the complication of extrapolation or interpolation of data. Since the velocity field is known at time-level $t - \Delta t$, this can be used in an approximation of a separate trajectory across two time-levels :

$$x(t - 2\Delta t) = x(t) - 2\Delta t\, u\left[\frac{1}{2}[x(t) + x(t - 2\Delta t)], t - \Delta t\right].$$

Rivest et al. compare weather simulations employing the great-circle extrapolation method with those obtained by using double trajectory calculations. (The piecewise trajectory method is not considered.) The extrapolation method exhibits slightly inferior accuracy, but has the advantage of computational speed.

11.1 Three-Time-Level Schemes

We shall now take the solution to the orographic resonance problem, described above, as a

Through a series of trial integrations, on a global shallow-water model, optimal values of and δ are deduced. A value of δ above 1/6 will su□ciently dampen resonance, but a value larger than 1/2 will lead to increased temporal truncation errors. It would appear that the choice of is less critical, except that it should be small and positive.

Summary

In this thesis we have examined recent developments in semi-Lagrangian schemes. The earliest uses of SL schemes in meteorological applications indicated deficiencies which have lately been addressed [55]. Among these we mention the computational cost of interpolation, the lack of formal conservation and the problem of orographic resonance.

Two approaches to reducing the computational cost of interpolation have been examined in this thesis. The noninterpolating methods discussed in Chapter 6 remove interpolation completely from SL schemes. Eulerian advection schemes are used in place of interpolation. The noninterpolating formalism therefore o□ers a framework for the development of many new schemes.

An alternative to more traditional forms of interpolation has been introduced by Purser and Lesley, as discussed in Chapter 5. This method significantly reduces the number of univariate interpolations required to produce a single multivariate interpolation, in comparison to standard tensor product methods. A further development of this method has incorporated conservation of mass. To date, it is not clear how this latter scheme may be implemented in spherical geometry. For this reason the method is not yet applicable to global atmospheric modelling.

Besides the issue of computational speed, further requirements of interpolation for SL schemes have been discussed. In Chapter 4 we saw how the introduction of non-physical errors may be avoided through the use of monotone, or shape-preserving interpolation.

the range of possible applications of the SL method.

Lastly, in Chapter 11 we examined the problem of spurious resonance caused by orography. In early SL models, where mountain topography was included at the lower boundary, spurious standing wave patterns presented a particular problem. A sequence of papers has lately addressed this problem, demonstrating further the suitability of semi-

Bibliography

[1] H. Akima. A new method of interpolation and smooth curve fitting based on local procedures. J. Assoc. Comput. Mach., 17:589–602, 1970.

[2] J.R. Bates, F.H.M. Semazzi, R.W. Higgins, and S.R.M. Barros. Integration of the shallow water equations on the sphere using a vector semi-Lagrangian scheme with a multigrid solver. MWR, 118:1615–1627, 1990.

[3] R. Bermejo and A. Staniforth. The conversion of semi-Lagrangian schemes to quasi-monotone schemes. Monthly Weather Review, 120:2622–2632, 1992.

[4] J.P. Boris and D.L. Book. Flux corrected transport, I, SHASTA, a fluid transport algorithm that works. Journal of Computational Physics, 11:38–69, 1973.

[5] J.C. Côté, S. Gravel, and A. Staniforth. A generalized family of schemes tht eliminate the spurious resonant response of semi-Lagrangian schemes to orographic forcing. Monthly Weather Review, 123:3605–3613, 1995.

[6] R. Courant, K.O. Friedrichs, and H. Lewy. On the partial di□erence equations of mathematical physics. Math. Annalen, 100:32–74, 1928. English translation by Phyllis Fox, IBM Jnl, March 1967.

[7] R.L. Dougherty, A. Edelman, and J.M. Hyman. Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic Hermite interpolation. Mathematics of Computation, 52:471–494, 1989.

[31] K.W. Morton and D.F. Mayers. Numerical Solution of Partial Di□erential Equations. Cambridge University Press, 1994.

[32] P. Garcia Navarro and A. Priestley. A conservative and shape-preserving semi-Lagrangian method for the solution of the shallow-water equations. International Journal for Numerical Methods in Fluids, 18:273–294, 1994.

[33] G. Nicolis. Introduction to Nonlinear Science. Cambridge University Press, 1995.

[34] M. Olim. A truly noninterpolating semi-Lagrangian Lax-Wendro□ method. Journal of Computational Physics, 112:253–266, 1994.

[35] A. Plante. Transport de Substances par Schémas Numériques Semi-Lagrangiens Intégrés par Cellules. PhD thesis, Physics Department, University of Québec at Montréal, 1993.

[36] P.M. Prenter. Spline and Variational Methods. 1975.

[37] A. Priestley. Private communication.

[38] A. Priestley. A quasi-conservative version of the semi-Lagrangian advection scheme. Monthly Weather Review, 121:621–629, 1993.

[39] R.J. Purser and L.M. Leslie. An e□cient interpolation procedure for high-order three-dimensional semi-Lagrangian models. Monthly Weather Review, pages 2492–2498, 1991.

[40] R.J. Purser and L.M. Leslie. An e□cient semi-Lagrangian scheme using third-order semi-implicit time integration and forward trajectories. Monthly Weather Re2tion p

pR.W3[illegible]L T2[illegible]di-c

[42] M. Rančić. An e□cient conservative, monotonic remapping for semi-Lagrangian transport algorithms. Monthly Weather Review, 123:1213–1217, 1995.

[43] P.J. Rasch and D.L. Williamson. On shape-preserving interpolation and semi-Lagrangian transport. SIAM J. Sci. Stat. Comp., 11, 1990.

[44] H. Ritchie. Eliminating the interpolation associated with the semi-Lagrangian scheme. Monthly Weather Review, 114:135–146, 1986.

[45] H. Ritchie and C. Beaudoin. Approximations and sensitivity experiments with a baroclinic semi-Lagrangian spectral model. Monthly Weather Review, 122:2391–2399, 1994.

[46] C. Rivest, A. Staniforth, and A. Robert. Spurious resonant response of semi-Lagrangian discretizations to orographic forcing : Diagnosis and solution.

[53] P.K. Smolarkiewicz and J.A. Pudykiewicz. A class of semi-Lagrangian approximations for fluids. Journal of the Atmospheric Sciences, 49:2082–2096, 1992.

[54] P.K. Smolarkiewicz and P.J. Rasch. Monotone advection on the sphere : An Eulerian versus semi-Lagrangian approach. Journal of the Atmospheric Sciences, 48:793–810, 1991.

[55] A. Staniforth and J. Côté. Semi-Lagrangian integration schemes for atmospheric models. Monthly Weather Review, 119:2206–2223, 1991.

[56] A. Staniforth, J. Côté, and J. Pudykiewicz. Comments on "Smolarkiewicz's deformational flow". Monthly Weather Review, 115:894–900, 1987.

[57] P.K. Sweby. High-resolution schemes using flux limiters for hyperbolic conservation laws. SIAM J. Numerical Analysis, 121:995–1011, 1984.

[58] M.C. Tapp and P.W. White. A non-hydrostatic mesoscale model. Quarterly Journal of the Royal Meteorological Society, 102:277–296, 1976.

[59] C. Temperton and A. Staniforth. An e□cient two-time-level semi-Lagrangian semi-implicit integration scheme. Quarterly Journal of the Royal Meteorological Society, 113:1025–1039, 1987.

[60] D.L. Williamson, J.B. Drake, J.J. Hack, R. Jakob, and P.N. Swarztrauber. A standard test set for numerical approximations to the shallow-water equations in spherical geometry. Journal of Computational Physics, 102:211–224, 1992.

[61] D.L. Williamson and P.J. Rasch. Two-dimensional semi-Lagrangian transport with shape-preserving interpolation. Monthly Weather Review, 117:102–129, 1989.

[62] S.T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. Journal of Computational Physics, 31:335–362, 1979.