Finite Difference Methods

Learning Outcomes

Following this lecture, students will be able to:

- Describe the differences between explicit and implicit temporal differencing methods.
- Describe the fundamental reasoning for using semi-implicit or split-explicit temporal differencing methods within modern numerical weather prediction models.
- Describe the differences between Eulerian and Lagrangian spatial differencing methods.
- Derive even- and odd-order-accurate Eulerian spatial finite differencing approximations.

Generalized Temporal Differencing Methods

There exist two basic types of temporal differencing methods: **explicit** and **implicit** methods.

With explicit methods, future-time variables appear only on one side of an equation written using finite differences. An example explicit finite difference is the centered-in-time finite difference approximation. Consider a simple 1-D linear advection equation for a generic variable *h*:

$$\frac{\partial h}{\partial t} = -u\frac{\partial h}{\partial x}$$

We wish to solve this equation at a generic time *t* and generic location *x*. Apply centered finite difference approximations in time and space to the partial derivatives above to obtain:

$$\frac{h_{t+1}^{x} - h_{t-1}^{x}}{2\mathit{\Delta} t} = -u_t^x\frac{h_t^{x+1} - h_t^{x-1}}{2\mathit{\Delta} x}$$

Subscripts indicate the time iteration for a given variable and superscripts indicate the grid location for a given variable. Δ*t* and Δ*x* indicate the timestep and horizontal grid spacing. Solving this equation for h_{t+1}^{x}, we obtain:

$$h_{t+1}^{x} = h_{t-1}^{x} - \mathit{\Delta} t u_t^x\frac{h_t^{x+1} - h_t^{x-1}}{\mathit{\Delta} x}$$

In other words, *h* at grid location *x* at time *t+1* is equal to *h* at grid location *x* at a previous time *t-1* **plus** the forcing at time *t* – here, the advection of *h* by *u* – multiplied by the timestep. The future time, *t+1,* appears only on the left-hand side of the equation, with current and past times appearing on the right-hand side of the equation.

By contrast, with **implicit** methods, variables at the future time appear on both sides of the equation. This means that they require using iterative methods to solve. For example, apply a backward finite difference in time and centered finite difference in space to the partial derivatives in the 1-D linear advection equation:

$$\frac{h_t^x - h_{t-1}^x}{\Delta t} = -u_t^x \frac{h_t^{x+1} - h_t^{x-1}}{2\Delta x}$$

Solving this equation for h_t^x, we obtain:

$$h_t^x = h_{t-1}^x - \Delta t u_t^x \frac{h_t^{x+1} - h_t^{x-1}}{2\Delta x}$$

In other words, *h* at the current time is equal to *h* at the previous time plus the forcing at the current time multiplied by the timestep. Because the time subscript *t* is generic, we can add 1 to each time subscript to write the equation as:

$$h_{t+1}^x = h_t^x - \Delta t u_{t+1}^x \frac{h_{t+1}^{x+1} - h_{t+1}^{x-1}}{2\Delta x}$$

Here, *h* at the future time is equal to *h* at the current time plus the forcing at the future time multiplied by the timestep. The only thing that changed is the time subscript. In this equation, h_{t+1} appears on both sides of the equation, and the forcing term is a function of *u* valid at the future time as well.

A generic form of this equation, where the forcing is represented by a generic function *G*, takes the form:

$$h_{t+1}^x = h_t^x + G_{t+1}^x \Delta t$$

where for this example, $G_{t+1}^x = -u_{t+1}^x \frac{h_{t+1}^{x+1} - h_{t+1}^{x-1}}{2\Delta x}$

Note that the specific form of *G* varies between spatial differencing schemes and applies to both explicit and implicit methods. For brevity, we will use *G* to represent the forcing throughout the rest of these notes.

Implicit temporal differencing schemes are typically *unconditionally stable*, such that the model timestep is unconstrained by a limiting value of the Courant number. In other words, *C* no longer must be less than or equal to 1 and can instead approach infinity. In theory, this allows for an infinitely long model timestep. Of course, in practice we do not use an infinitely long model timestep: we know that the forcing changes over time and, as a result, we wish to use a model timestep that maximizes computational efficiency (longer than when Δ*t* is constrained) yet also maximizes computational accuracy (short enough to reflect variability in the forcing term(s)).

However, as noted before, implicit temporal differencing schemes require iterative methods to solve. Such methods are computationally expensive, with the added computational expense from iterative methods often outweighing the computational benefit from a longer timestep. Instead of applying implicit temporal differencing methods to *all* terms in the primitive equations, however, we can instead apply implicit temporal differencing methods to a *subset* of terms. Such methods are known as **semi-implicit** methods.

With semi-implicit methods, implicit temporal differencing is applied to terms for which a long timestep is beneficial and explicit temporal differencing is applied to all other terms. For example, recall that the Courant number is defined as a function of the speed *U* of the fastest wave on the model grid. For fully compressible models, which permit sound (or acoustic) waves, this is the speed of sound (~340 m s^{-1}). Thus, semi-implicit methods typically handle sound waves implicitly and everything else explicitly.

As an example, consider the full *u*-momentum equation, in Cartesian coordinates and with *z* as the vertical coordinate as we did earlier in the semester:

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - w\frac{\partial u}{\partial z} + fv - \frac{1}{\rho}\frac{\partial p}{\partial x}$$

Here, we have neglected the curvature and frictional terms. We have also applied a scale analysis to neglect the $2\Omega w\cos\phi$ component to the Coriolis force. Since sound waves are associated with changes in pressure in response to compression or expansion (i.e., changes in density), we wish to implicitly represent the horizontal pressure gradient term in the above equation.

We can do so by splitting the equation by the terms to be solved explicitly and implicitly:

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - w\frac{\partial u}{\partial z} + fv \qquad \text{solved explicitly}$$

$$\frac{\partial u}{\partial t} = -\frac{1}{\rho}\frac{\partial p}{\partial x} \qquad \text{solved implicitly}$$

After each set of forcings is computed, the results are added together to advance the model.

Explicit numerical diffusion terms, or artificial damping terms added to the model’s equations to reduce feature amplitudes (specifically those at very-short wavelengths), are also typically solved implicitly. This is because the coefficients on these artificial damping terms that control how much to dampen the solution at each time step are often so large as to result in computational instability if solved using explicit methods.

The advantage of semi-implicit methods is that terms that would otherwise need a short timestep to maintain computational stability, such as sound waves and explicit numerical diffusion, can be handled with a longer timestep. It also allows us to use a timestep that is a function of the prevailing meteorology for all other terms. Both the GFS and ECMWF models utilize semi-implicit temporal differencing methods.

There are also **split-explicit**, or *time-splitting*, methods. These methods are fully explicit, but use short timesteps for forcing terms that are associated with sound waves and longer timesteps for the remaining terms. As with semi-implicit methods, the time tendencies from each set of terms are added together at the end of the longer timestep to obtain the total time tendency. Split-explicit methods are used in modern NWP models such as WRF-ARW and MPAS. A practical example for the WRF-ARW model is given by Fig. 3.1 of Skamarock et al. (2008).

More on Explicit Temporal Differencing Methods

Above, we introduced the “leapfrog” centered-in-time finite differencing method as an example of an explicit temporal differencing method. A forward-in-time finite differencing method can also be used, and this too is an example of an explicit temporal differencing method.

Consider again a simple 1-D linear advection equation for a generic variable *h*. Applying a forward finite difference in time and representing the advection term as *G*, we obtain:

$$h_{t+1}^{x} = h_{t}^{x} + \mathit{\Delta} t G_{t}^{x}$$

where $G_t^x = -u_t^x \frac{h_t^{x+1} - h_t^{x-1}}{2\mathit{\Delta} x}$ for a centered finite difference in space.

However, these are not the only explicit methods that exist. Another explicit method is the Adams-Bashforth method, which has the generic form:

$$h_{t+1}^{x} = h_{t}^{x} + \frac{\mathit{\Delta} t}{12}\left(23G_{t}^{x} + 16G_{t-1}^{x} + 5G_{t-2}^{x}\right)$$

Here, the value of h_{t+1}^{x} depends on the forcing at the current time *t* and at the previous two times *t-1* and *t-2*. Substituting the definition for *G* into this expression and expanding, we obtain:

$$h_{t+1}^{x} = h_{t}^{x} - \frac{\mathit{\Delta} t}{24\mathit{\Delta} x}\left(23u_{t}^{x}\left(h_{t}^{x+1} - h_{t}^{x-1}\right) + 16u_{t-1}^{x}\left(h_{t-1}^{x+1} - h_{t-1}^{x-1}\right) + 5u_{t}^{x}\left(h_{t-2}^{x+1} - h_{t-2}^{x-1}\right)\right)$$

This expression is somewhat similar to the forward-in-time scheme since the total forcing is 23/24ths of that valid at time *t*, but it also includes partial contributions from the forcing at times *t-1* and *t-2*. The forward-in-time, leapfrog, and Adams-Bashforth differencing schemes each have their own distinct accuracy and stability characteristics, as do all temporal differencing schemes, and we will explore these concepts further for selected schemes later in this and subsequent lectures.

There are also *predictor-corrector* explicit methods, which use two or more steps to advance to the next forecast time. Though they involve greater computational expense because they require two or more times as many calculations to advance between forecast times, they tend to have superior accuracy and/or stability characteristics that make this trade-off acceptable.

One predictor-corrector scheme is the two-step Lax-Wendroff scheme:

$$h_{t*}^{x} = \frac{h_t^{x+1} - h_t^{x-1}}{2} + \frac{\Delta t}{2} G_t^{x} \qquad \text{(predictor step)}$$

$$h_{t+1}^{x} = h_t^{x} + \Delta t G_{t*}^{x} \qquad \text{(corrector step)}$$

The value of *h* is first evaluated at an intermediate time *t** (*t* < *t** < *t*+1). Subsequently, the value of *h* is evaluated at the next time *t*+1 using the forcing from the intermediate time but also using the full timestep Δ*t*.

Another two-step predictor-corrector scheme is the Euler-backward or Matsuno scheme:

$$h_{t*}^{x} = h_t^{x} + \Delta t G_t^{x} \qquad \text{(predictor step)}$$

$$h_{t+1}^{x} = h_t^{x} + \Delta t G_{t*}^{x} \qquad \text{(corrector step)}$$

For this scheme, the predictor step is akin to a forward-in-time finite difference except that it is valid at an intermediate time *t**. This predicted value is then used to compute the forcing *G* at time *t**, from which the value of *u* at the next time *t*+1 can be obtained.

It is also possible to devise and use three-step predictor-corrector schemes. A popular three-step scheme is the Runge-Kutta 3 (or RK3) scheme, which is used by both the WRF-ARW and MPAS models. This takes the generic form:

$$h_{t*}^{x} = h_t^{x} + \frac{\Delta t}{3} G_t^{x} \qquad \text{(first predictor step)}$$

$$h_{t'}^{x} = h_t^{x} + \frac{\Delta t}{2} G_{t*}^{x} \qquad \text{(second predictor step)}$$

$$h_{t+1}^{x} = h_t^{x} + \Delta t G_{t'}^{x} \qquad \text{(corrector step)}$$

Here, there are two intermediate times, *t** and *t'*, at which *h* is computed and subsequently used to compute the forcing *G* at those times. The result is the value of *h* at time *t*+1. There is also a general class of Runge-Kutta multistep schemes that include two-step (resembling the last two steps of the RK3 scheme) and four-step schemes.

Eulerian Spatial Differencing Methods

There are two primary classes of spatial differencing methods: **Eulerian** and **Lagrangian**. Eulerian methods are *fixed to the model grid*, whereas Lagrangian methods are *flow-following*. We first wish to consider Eulerian methods, as are used by most modern numerical weather prediction models including the WRF-ARW model.

As we did earlier in this lecture, consider a simplified form of the *u*-momentum equation:

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - w\frac{\partial u}{\partial z} + fv - \frac{1}{\rho}\frac{\partial p}{\partial x}$$

With Eulerian methods, all forcing terms are evaluated at each model grid point at each time. In this case, the local time rate of change of *u* at each grid point is equal to the advection of *u* by the three-dimensional wind plus additional forcing terms, given here by the Coriolis and horizontal pressure gradient terms. The chosen timestep must adhere to the CFL stability criterion.

Recall that the partial derivative of a generic function *f* with respect to a generic variable *x* can be expressed as:

$$\frac{\partial f}{\partial x} = \lim_{\Delta x \to 0} \frac{\Delta f}{\Delta x} \tag{1}$$

We desire to evaluate (1) at a point x_a. Using Taylor series, we can expand *f(x)* about two points on either side of x_a: x_{a+1} and x_{a-1}. These expansions take the form:

$$f(x_{a+1}) = f(x_a) + \Delta x\frac{\partial f}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 f}{\partial x^2} + \frac{(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3} + \frac{(\Delta x)^4}{4!}\frac{\partial^4 f}{\partial x^4} + \frac{(\Delta x)^5}{5!}\frac{\partial^5 f}{\partial x^5} + \ldots \tag{2}$$

$$f(x_{a-1}) = f(x_a) - \Delta x\frac{\partial f}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 f}{\partial x^2} - \frac{(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3} + \frac{(\Delta x)^4}{4!}\frac{\partial^4 f}{\partial x^4} - \frac{(\Delta x)^5}{5!}\frac{\partial^5 f}{\partial x^5} + \ldots \tag{3}$$

The forward finite difference approximation is obtained by solving (2) for $\frac{\partial f}{\partial x}$. The backward finite difference approximation is obtained by solving (3) for $\frac{\partial f}{\partial x}$. The centered finite difference approximation is obtained by subtracting (3) from (2) and solving for $\frac{\partial f}{\partial x}$. These approximations are said to be first-, first-, and second-order accurate, respectively. We can illustrate this using the centered finite difference approximation. Subtract (3) from (2) to obtain:

$$f(x_{a+1}) - f(x_{a-1}) = 2\Delta x\frac{\partial f}{\partial x} + \frac{2(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3} + \frac{2(\Delta x)^5}{5!}\frac{\partial^5 f}{\partial x^5} + \ldots \tag{4}$$

Solving for $\frac{\partial f}{\partial x}$, we obtain:

$$\frac{\partial f}{\partial x} = \frac{f(x_{a+1}) - f(x_{a-1})}{2\Delta x} - \frac{(\Delta x)^2}{3!}\frac{\partial^3 f}{\partial x^3} - \frac{(\Delta x)^4}{5!}\frac{\partial^5 f}{\partial x^5} + \ldots \tag{5}$$

Truncating (5) after the first term on the right-hand side of the equation results in the second-order centered finite difference approximation. The error in this approximation is equal to the sum of the truncated terms, the largest of which involves $(\Delta x)^2$. The order of accuracy of the finite difference approximation is equal to the exponent on Δx of the first (largest) truncated term. Here, this is 2, such that the approximation is second-order accurate.

One can obtain higher-order accurate finite difference approximations so long as the function *f* is continuously differentiable (i.e., if higher-order partial derivatives of *f* exist). For instance, fourth-order and sixth-order centered approximations can be developed. To obtain the fourth-order difference, we start with (5). Replace Δx with $2\Delta x$, which necessitates replacing x_{a+1} with x_{a+2} and x_{a-1} with x_{a-2}, to obtain:

$$\frac{\partial f}{\partial x} = \frac{f(x_{a+2}) - f(x_{a-2})}{2(2\Delta x)} - \frac{(2\Delta x)^2}{3!}\frac{\partial^3 f}{\partial x^3} - \frac{(2\Delta x)^4}{5!}\frac{\partial^5 f}{\partial x^5} + \ldots \tag{6}$$

Next, multiply (5) by 2^2. If we subtract that result from (6) and simplify, we obtain:

$$\frac{\partial f}{\partial x} = \frac{8(f(x_{a+1}) - f(x_{a-1})) - (f(x_{a+2}) - f(x_{a-2}))}{12\Delta x} + \frac{4(\Delta x)^4}{5!}\frac{\partial^5 f}{\partial x^5} + \ldots \tag{7}$$

Note how the $(\Delta x)^2$ terms in (6) and from (5) no longer appear. This leaves the $(\Delta x)^4$ term as the lowest-order term beyond the first term in (7). Truncating this and all higher-order terms results in the fourth-order accurate centered finite difference approximation. It depends on the value of *f* at four points: x_{a-2}, x_{a-1}, x_{a+1}, and x_{a+2}.

The process of obtaining the sixth-order accurate finite difference approximation is similar. Start with (5) and replace Δx with $3\Delta x$ (and thus x_{a+3} with x_{a+3} and x_{a-3} with x_{a-3}). Multiply (5) by 3^2 and subtract it from the just-obtained equation to obtain an equation in which the $(\Delta x)^2$ terms have been eliminated. Next, multiply this equation by 2^2 and subtract the result from (7) multiplied by 3^2 to obtain an equation in which the $(\Delta x)^4$ terms have been eliminated:

$$\begin{aligned}\frac{\partial f}{\partial x} = {} & \frac{45(f(x_{a+1}) - f(x_{a-1})) - 9(f(x_{a+2}) - f(x_{a-2})) + (f(x_{a+3}) - f(x_{a-3}))}{60\Delta x} \\ & - \frac{36(\Delta x)^6}{7!}\frac{\partial^7 f}{\partial x^7} + \ldots\end{aligned} \tag{8}$$

Truncating the $(\Delta x)^6$ and higher-order terms results in the sixth-order accurate finite difference approximation. It depends on the value of *f* at six points: x_{a-3}, x_{a-2}, x_{a-1}, x_{a+1}, x_{a+2}, and x_{a+3}. Even-higher-order – and thus more accurate –approximations may be developed in a similar manner to the fourth- and sixth-order accurate formulations described above.

Odd-order accurate (third-order, fifth-order, etc.) approximations are also possible. Like the first-order accurate forward and backward approximations, these odd-order accurate approximations are not centered. Instead, they involve a larger number of points on one side of x_a than on the other side.

To derive a third-order accurate, upwind-biased (e.g., involving more points on the negative side of x_a than on the positive side) finite difference approximation, let us step back to first principles. Although we have conceptualized a differencing scheme's order of accuracy as being equal to the exponent on the first Δ*x* term truncated from the expression, it is also equal to one less than the number of grid points involved in calculating the approximation. This is sometimes referred to as a calculation *stencil*. The second-order accurate centered finite difference uses three points: x_a, x_{a+1}, and x_{a-1}. Similarly, the third-order accurate upward finite difference uses four points: x_a, x_{a+1}, x_{a-1}, and x_{a-2}.

Let us start with the Taylor series expansions of *f* at each of these points. We already have the expansions for the *a+1* and *a-1* points; these are given by (2) and (3) above. The expansion for the *a* point is pretty straightforward: all of the *x – b* terms are zero, such that we get the trivial result:

$$f(x_a) = f(x_a) \tag{9}$$

which can be thought of as having higher-order partial derivative terms with coefficients of zero.

The expansion for the *a-2* point is not particularly challenging, either. Here, *x-b* = -2Δ*x*. Thus:

$$\begin{aligned} f(x_{a-2}) = f(x_a) - 2\Delta x\frac{\partial f}{\partial x} + \frac{4(\Delta x)^2}{2!}\frac{\partial^2 f}{\partial x^2} - \frac{8(\Delta x)^3}{3!}\frac{\partial^3 f}{\partial x^3} + \frac{16(\Delta x)^4}{4!}\frac{\partial^4 f}{\partial x^4} \\ - \frac{32(\Delta x)^5}{5!}\frac{\partial^5 f}{\partial x^5} + \ldots \end{aligned} \tag{10}$$

Divide (2), (3), (9), and (10) by Δ*x*, drop the fifth partial derivatives and higher-order terms, and expand the factorials to obtain:

$$\frac{f(x_{a+1})}{\Delta x} = \frac{f(x_a)}{\Delta x} + \frac{\partial f}{\partial x} + \frac{(\Delta x)}{2}\frac{\partial^2 f}{\partial x^2} + \frac{(\Delta x)^2}{6}\frac{\partial^3 f}{\partial x^3} + \frac{(\Delta x)^3}{24}\frac{\partial^4 f}{\partial x^4} + \ldots \tag{11a}$$

$$\frac{f(x_a)}{\Delta x} = \frac{f(x_a)}{\Delta x} \tag{11b}$$

$$\frac{f(x_{a-1})}{\Delta x} = \frac{f(x_a)}{\Delta x} - \frac{\partial f}{\partial x} + \frac{(\Delta x)}{2}\frac{\partial^2 f}{\partial x^2} - \frac{(\Delta x)^2}{6}\frac{\partial^3 f}{\partial x^3} + \frac{(\Delta x)^3}{24}\frac{\partial^4 f}{\partial x^4} + \ldots \tag{11c}$$

$$\frac{f(x_{a-2})}{\Delta x} = \frac{f(x_a)}{\Delta x} - 2\frac{\partial f}{\partial x} + \frac{4(\Delta x)}{2}\frac{\partial^2 f}{\partial x^2} - \frac{8(\Delta x)^2}{6}\frac{\partial^3 f}{\partial x^3} + \frac{16(\Delta x)^3}{24}\frac{\partial^4 f}{\partial x^4} + \ldots \tag{11d}$$

To obtain the third-order accurate finite difference approximation for the first derivative, we need to manipulate these equations so that the coefficients on $\frac{\partial f}{\partial x}$ sum to 1 and the coefficients on $\frac{f(x_a)}{\Delta x}$, $\frac{(\Delta x)}{2!}\frac{\partial^2 f}{\partial x^2}$, and$\frac{(\Delta x)^2}{3!}\frac{\partial^3 f}{\partial x^3}$ sum to 0 when the equations are added together. This leaves the fourth partial derivative term with a leading $(\Delta x)^3$ as the first truncated term, giving the desired third-order accurate result. This can be expressed using matrix multiplication:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & -2 \\ \frac{1}{2} & 0 & \frac{1}{2} & 2 \\ \frac{1}{6} & 0 & -\frac{1}{6} & -\frac{8}{6} \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

The first, second, third, and fourth rows of the 4x4 matrix are the leading factors on $\frac{f(x_a)}{\Delta x}$, $\frac{\partial f}{\partial x}$, $\frac{(\Delta x)}{2!}\frac{\partial^2 f}{\partial x^2}$, and $\frac{(\Delta x)^2}{3!}\frac{\partial^3 f}{\partial x^3}$ in (11a-d), respectively. When multiplied by the column coefficient matrix, the first, third, and fourth rows must sum to zero whereas the second row must sum to 1, giving the column result matrix on the right-hand side.

The above matrices take the form **Bx** = **C**. However, it is **x** that is the unknown matrix, such that we must instead solve for **x** = $\mathbf{B}^{-1}\mathbf{C}$. Using an automated routine, we find that:

$$\mathbf{B}^{-1} = \begin{bmatrix} 0 & \frac{1}{3} & 1 & 1 \\ 1 & \frac{1}{2} & -2 & -3 \\ 0 & -1 & 1 & 3 \\ 0 & \frac{1}{6} & 0 & -1 \end{bmatrix}$$

Thus, our equation becomes:

$$\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{3} & 1 & 1 \\ 1 & \frac{1}{2} & -2 & -3 \\ 0 & -1 & 1 & 3 \\ 0 & \frac{1}{6} & 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Performing the matrix multiplication on the right-hand side of this equation, we find that a = 1/3, b = 1/2, c = -1, and d = 1/6. In other words, multiplying (11a) by 1/3, (11b) by 1/2, (11c) by -1, and (11d) by 1/6 and adding them together eliminates the $\frac{f(x_a)}{\Delta x}$, $\frac{(\Delta x)}{2!}\frac{\partial^2 f}{\partial x^2}$, and $\frac{(\Delta x)^2}{3!}\frac{\partial^3 f}{\partial x^3}$ terms, leaving an expression for our first partial derivative:

$$\frac{1}{3}\left(\frac{f(x_{a+1})}{\Delta x}\right) + \frac{1}{2}\left(\frac{f(x_a)}{\Delta x}\right) - \left(\frac{f(x_{a-1})}{\Delta x}\right) + \frac{1}{6}\left(\frac{f(x_{a-2})}{\Delta x}\right) = \frac{\partial f}{\partial x} \qquad (12)$$

Or, with a common denominator of 6Δ*x*:

$$\frac{\partial f}{\partial x} = \left(\frac{2f(x_{a+1}) + 3f(x_a) - 6f(x_{a-1}) + f(x_{a-2})}{6\mathit{\Delta} x} \right) \tag{13}$$

This result is upwind-biased: note the increased weighting on x_{a-1} relative to x_{a+1}, as well as the presence of the x_{a-2} point but not the x_{a+2} point.

The odd-order accurate finite difference approximations *implicitly numerically dampen* features. In other words, they cause a feature's amplitude to become smaller over time. Although we will discuss dampening in a later lecture, it is useful for us to demonstrate why this is the case. For instance, consider the following equation:

$$\frac{\partial f}{\partial x} = \frac{8(f(x_{a+1}) - f(x_{a-1})) - (f(x_{a+2}) - f(x_{a-2}))}{12\mathit{\Delta} x} + \frac{2(\mathit{\Delta} x)^3}{4!}\frac{\partial^4 f}{\partial x^4} \tag{14}$$

The first right-hand side term of (14) is the fourth-order accurate approximation. The second right-hand side term is a fourth partial derivative. If we use a centered finite difference approximation for this fourth partial derivative, we can write (14) as:

$$\begin{aligned} \frac{\partial f}{\partial x} = &\frac{8(f(x_{a+1}) - f(x_{a-1})) - (f(x_{a+2}) - f(x_{a-2}))}{12\mathit{\Delta} x} \\ &+ \frac{2(\mathit{\Delta} x)^3}{24}\left(\frac{f(x_{a-2}) - 4f(x_{a-1}) + 6f(x_a) - 4f(x_{a+1}) + f(x_{a+2})}{(\mathit{\Delta} x)^4} \right) \end{aligned} \tag{15}$$

Simplifying the second right-hand side term of (15), we obtain:

$$\begin{aligned} \frac{\partial f}{\partial x} = &\frac{8(f(x_{a+1}) - f(x_{a-1})) - (f(x_{a+2}) - f(x_{a-2}))}{12\mathit{\Delta} x} \\ &+ \frac{f(x_{a-2}) - 4f(x_{a-1}) + 6f(x_a) - 4f(x_{a+1}) + f(x_{a+2})}{12\mathit{\Delta} x} \end{aligned} \tag{16}$$

Combining like terms, we are left with 2*f*(x_{a-2}), -12*f*(x_{a-1}), 6*f*(x_a), 4*f*(x_{a+1}), and 0*f*(x_{a+2}):

$$\begin{aligned} \frac{\partial f}{\partial x} &= \frac{4f(x_{a+1}) + 6f(x_a) - 12f(x_{a-1}) + 2f(x_{a-2})}{12\mathit{\Delta} x} \\ &= \frac{2f(x_{a+1}) + 3f(x_a) - 6f(x_{a-1}) + f(x_{a-2})}{6\mathit{\Delta} x} \end{aligned} \tag{17}$$

where the second equivalency on the right is obtained by dividing the first by 2. We find that (17) is equal to (13). This means that (13) and (14) are equivalent ways of writing the third-order accurate upwind-biased finite difference approximation.

Why is this useful? As we will demonstrate in a later lecture, all even-order partial derivatives are dampening terms. Think of this in terms of the second partial derivative: we know that it is large where a quantity is locally small and vice versa. Thus, if you are computing the time tendency of a quantity, as we generally are in numerical weather prediction, adding a second partial derivative makes larger something that is locally small and makes smaller something that is locally large. The same applies to the fourth, sixth, and higher-order accurate even-exponent partial derivatives. Thus, (14) illustrates that the third-order accurate upwind-biased finite difference scheme includes a dampening term, whereas this term is hidden by the simpler formulation given by (13).

Lagrangian and Semi-Lagrangian Spatial Differencing Methods

With Lagrangian methods, the simplified form of the *u*-momentum equation is instead written in terms of the total derivative of *u*:

$$\frac{Du}{Dt} = fv - \frac{1}{\rho}\frac{\partial p}{\partial x}$$

Here, the change in *u* following the motion is equal to the sum of the Coriolis and horizontal pressure gradient forcing terms, also evaluated following the motion. The advection terms are encapsulated within the definition of the total derivative; consequently, the chosen timestep is not limited by the CFL stability criterion. Lagrangian methods also mitigate the deleterious effects of aliasing, non-linear instability, and truncation error, all concepts that we have discussed or will discuss later this semester. Consequently, Lagrangian methods are an appealing option for handling spatial differencing.

A model simulation conducted utilizing Lagrangian methods might begin with a structured grid. Air parcels at each model grid point are then followed forward in time. The flow-following nature of the method results in air parcels (and thus model grid "points") clustering in areas of convergent flow and away from areas of divergent flow. This is not ideal; areas of divergent flow (synoptic-scale anticyclones, thunderstorm downdrafts, etc.) quickly become and often remain poorly resolved, and the grid becomes irregularly structured. Further, model quantities that should be conserved by a simulation do not always remain conserved using these methods.

An alternative to a purely Lagrangian method is a **semi-Lagrangian** method (Fig. 1), which are used by the GFS and ECMWF models. Instead of tracking a single set of air parcels throughout the entire model simulation, a new set of air parcels is defined with each timestep. This mitigates the parcel clustering issues of a purely Lagrangian method (though not its conservation issues) while not detracting from its positive attributes.

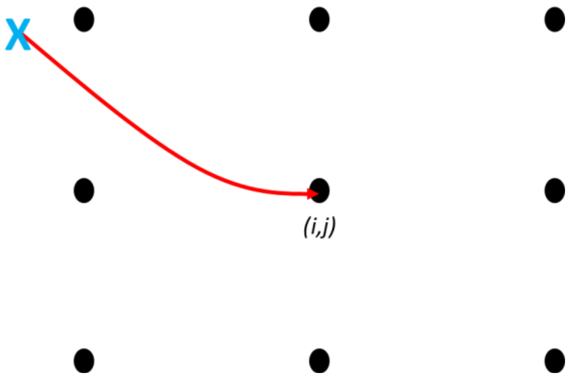

Figure 1. Schematic of a backward semi-Lagrangian method on a structured model grid. Here, model variables on this grid are known at time *t*. We wish to obtain the values of model variables on this grid at time *t+1*. Backward trajectories are released at time *t+1* from each model grid point (*i*,*j*), from which their position at time *t* (**X**) may be obtained. Please refer to the text below for more details regarding the practical implementation of this method.

The practical implementation of semi-Lagrangian methods is as follows. Consider a model forecast at time *t*. For simplicity, we state that the model uses a structured grid on which the values of model variables are known at time *t*. We wish to obtain the values of model variables on this grid at the next time, *t+1*. To do so, we define a set of air parcels at each model grid point at the next time *t+1*. We then follow these air parcels *backward* in time for one timestep, to determine where on the model grid they originated.

Let the position of an air parcel at time *t+1* be **x**, representing the three-dimensional position vector. Let the position of this air parcel at time *t* be **x** – **α**, where **α** represents the air parcel's three-dimensional displacement vector. The value for **α** is given by:

$$\vec{\alpha} = \varDelta t^{*}\mathbf{v}(\mathbf{x} - \vec{\alpha}, t)$$

In other words, **α** is equal to the timestep multiplied by the velocity valid at position **x** – **α** and time *t*. As we do not know initially know the air parcel's location at time *t*, this equation is solved using iterative means. The first guess for **α** is then given by $\vec{\alpha} = \varDelta t^{*}\mathbf{v}(\mathbf{x}, t)$, which is then updated iteratively until the solution converges within a desired accuracy threshold. For small Δ*t*, a small number of iterations – e.g., two or three – may be needed to obtain **α**. Once **α** is known, so too is **x** – **α**, and thus the position of the air parcel at time *t* is known.

It is unlikely that **x** – **α** is located on a model grid point; rather, it is likely located between model grid points in three dimensions. This necessitates interpolation from model grid points at time *t* to each trajectory's location. While simple bilinear interpolation may be utilized to accomplish this, more advanced interpolation methods (e.g., cubic interpolation) are preferred for accuracy.

If the model variable is conserved following the motion, no further evaluation is needed. If the model variable is not conserved following the motion, the forcing *F* along the trajectory must be evaluated. One such way of representing this forcing is given by:

$$F = \frac{1}{2}\left[F(\mathbf{x}, t+1) + F(\mathbf{x} - \vec{\alpha}, t)\right]$$

In other words, the forcing *F* along the trajectory is simply the average of the forcing at the initial time *t* and position **x** – **α** and the forcing at the future time *t+1* and position **x**. Just as in the method above, however, iterative methods are required to obtain this forcing given that the value of *F* at (**x**, *t+1*) is not initially known.