Finite Difference Methods, Grid Staggering, and Truncation Error

Introduction to Temporal Differencing Methods

There exist two basic types of temporal differencing methods: **explicit** and **implicit** methods. With explicit methods, a model prognostic equation is approximated using finite differences in such a way that variables at the future time appear only on one side of an equation. An example explicit finite difference is that given by 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:

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

Subscripts indicate the time step for a given variable and superscripts indicate the grid location for a given variable, while Δt and Δx indicate the time step and horizontal grid spacing, respectively. Solving this equation for h_{t+1}^{x}, we obtain:

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

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 time step. This is an example of the "leapfrog" time differencing scheme we briefly considered earlier in the semester. We'll consider explicit methods in more detail shortly.

By contrast, with implicit methods, variables at the future time appear on both sides of the equation and require the use of 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 time step.

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

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

$$\text{where for this example, } G_t^x = -u_t^x \frac{h_t^{x+1} - h_t^{x-1}}{2\Delta x}$$

Because the time subscript *t* is generic, the generic equation can equivalently be written as:

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

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

Here, *h* at the 'current' time is still equal to *h* at the 'previous' time plus the forcing at the 'current' time multiplied by the time step. 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.

Implicit temporal differencing schemes are typically unconditionally stable, such that the model time step is unconstrained by some 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 time step. Of course, in practice we do not utilize such an infinitely long model time step – we know that the forcing on the variable will change over time and, as a result, we wish to use a model time step that maximizes computational efficiency (longer than when Δt 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 time step. Instead of applying implicit temporal differencing methods to *all* terms within 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 time step 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, this is the speed of sound (~340 m s^{-1}). Thus, semi-implicit methods typically handle sound wave terms 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. In this equation, the term that we wish to represent implicitly is the pressure gradient term, recalling that sound waves are associated with changes in pressure in response to compression or expansion (e.g., changes in density).

We can do so by considering the following two equations:

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

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

We have split the tendencies for the explicitly-solved forcing terms from those for the implicitly-solved forcing terms. After each set of forcing terms is computed, the results are added together to advance the model forward.

The advantage of semi-implicit methods is that terms that would otherwise need a short time step to maintain computational stability, such as sound waves, can be handled with a longer time step. We can then use a time step that is a function of the prevailing meteorology for all other terms. Both the GFS and ECMWF models utilize semi-implicit temporal differencing methods.

Explicit and Split-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 a generic function *G*, we obtain:

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

where $G_t^x = -u_t^x \dfrac{h_t^{x+1} - h_t^{x-1}}{2\Delta x}$ 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:

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

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:

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

In this form, it is clear that this expression is somewhat similar to the forward-in-time scheme since the total forcing is 23/24ths of that valid at time *t*. 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 also what are known as *predictor-corrector* explicit methods, which use two or more steps to advance to the next forecast time. Though they involve greater computational expense given that 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 time step Δt.

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:

$$h^x_{t^*} = h^x_t + \frac{\Delta t}{3} G^x_t \qquad \text{(first predictor step)}$$

$$h^x_{t'} = h^x_t + \frac{\Delta t}{2} G^x_{t^*} \qquad \text{(second predictor step)}$$

$$h^x_{t+1} = h^x_t + \Delta t G^x_{t'} \qquad \text{(corrector step)}$$

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 end result is the value of *h* at time *t*+1. There is also a general class of Runge-Kutta multistep schemes that has a four-step scheme as its basis. The two-step version (RK2) resembles the last two steps of the RK3 scheme.

It is also possible to devise what are known as **split-explicit**, or *time-splitting*, methods. Recall again that sound waves and, to some extent, gravity waves move rapidly, necessitating the use of a small time step to maintain computational stability. It is computationally expensive to use short time steps for all forcing terms, however. Instead, split-explicit methods use short time steps only for those terms that are associated with sound (and/or gravity) waves and longer time steps for the remaining terms. As with semi-implicit methods, the time tendencies from each set of terms can be added together at the end of the longer time step 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 Figure 3.1 of Skamarock et al. (2008).

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 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 time step must adhere to the CFL stability criterion. The WRF-ARW model is an example of a modern model that utilizes Eulerian methods.

What, then, are the Eulerian methods by which the partial derivatives in the above equation are evaluated? 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; e.g., 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* uniquely exist. While odd-order accurate (third-order, fifth-order, etc.) approximations are possible, they are not centered but instead involve a larger number of points on one side of x_a than on the other side. This is similar to the first-order accurate forward and backward finite difference approximations, with more points ahead of and behind x_a, respectively.

Fourth-order and sixth-order centered approximations can be developed, with the latter widely used for spatial derivatives in modern grid-point models. 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:

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

Next, multiply (5) by 2^2. If we subtract it from (6) and simplify, we can eliminate the $(\Delta x)^2$ terms:

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

Truncating the $(\Delta x)^4$ term results in the fourth-order accurate centered finite difference approximation. It depends upon 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. If this new equation is then multiplied by 2^2 and subtracted from (7)*3^2, an equation in which the $(\Delta x)^4$ terms have been eliminated is obtained:

$$\frac{\partial f}{\partial x} = \frac{45\left(f(x_{a+1}) - f(x_{a-1})\right) - 9\left(f(x_{a+2}) - f(x_{a-2})\right) + \left(f(x_{a+3}) - f(x_{a-3})\right)}{60\Delta x} - \frac{36(\Delta x)^6}{7!}\frac{\partial^7 f}{\partial x^7} + \ldots \qquad (8)$$

Truncating the $(\Delta x)^6$ term results in the sixth-order accurate finite difference approximation. It depends upon 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}. Higher-order – and thus more accurate – finite difference approximations may be developed similarly, the essence of which is to eliminate progressively higher-ordered partial derivatives from the Taylor series expansion.

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 time step is not limited by the CFL stability criterion. Lagrangian methods also mitigate the deleterious effects of aliasing, non-linear instability, and truncation error; we will discuss the first two of these concepts in more detail later in the 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 generally remain, poorly resolved. This also poses a challenge with respect to computing forcing terms involving partial derivatives, such as the horizontal pressure gradient term above, as the grid becomes irregular.

An alternative to the purely Lagrangian method is the **semi-Lagrangian** method (e.g., Figure 1). Instead of tracking a single set of air parcels throughout the entire model simulation, a new set of air parcels is defined with each time step. This mitigates the disadvantages of the Lagrangian method discussed above while not detracting from its positive attributes. However, one shortcoming of semi-Lagrangian (and Lagrangian) methods is that they typically do not conserve quantities evaluated using such methods. Despite this, the semi-Lagrangian method is presently widely used, including in the GFS and ECMWF models. Note that semi-Lagrangian methods can be used with both spectral and grid-based models.

The practical implementation of semi-Lagrangian methods is as follows. Here, we will consider the semi-Lagrangian equivalent to the forward-in-time differencing scheme, although a semi-Lagrangian equivalent to the centered-in-time leapfrog scheme can also be utilized. Consider a model forecast at time *t*. For simplicity, we state that the model uses a structured grid upon 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 wish to follow these air parcels *backward* in time for one time step, to determine where on the model grid they originated.

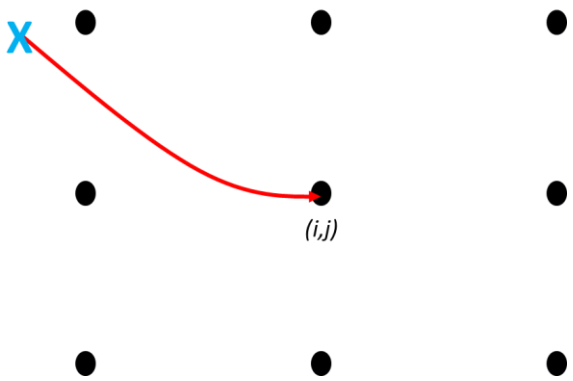


Figure 1. Schematic of a backward semi-Lagrangian method on a structured model grid. Here, model variables on this grid are known at time *t*. We wish to obtain the values of model variables on this grid at time *t+1*. Backward trajectories are released at time *t+1* from each model grid point (*i*,*j*), 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.

Once the air parcel positions at time *t* are known, interpolation from the structured grid at that time to the trajectory points at time *t* is used to obtain the values of the model variables. If a given model variable is conserved following the motion, the obtained value is then assigned to the trajectory's originating grid location at time *t+1*. If a given model variable is not conserved following the motion, then the value from time *t* is modified by the forcing terms applicable along the trajectory (e.g., the horizontal pressure gradient force term in the *u*-momentum equation), with the updated value then assigned to the trajectory's origin at time *t+1*.

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:

$$\vec{\alpha} = \Delta t * \mathbf{v}\left(\mathbf{x} - \vec{\alpha}, t\right)$$

In other words, **α** is equal to the time step 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 $\vec{\alpha} = \Delta t * \mathbf{v}\left(\mathbf{x}, t\right)$, 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.

Grid Staggering

Model variables may be defined on what are known as **staggered grids**, wherein mass-related fields such as pressure and temperature are defined on one set of grid points while velocity fields are defined on another set of grid points. Both horizontal and vertical grids may be staggered.

Staggered grids allow for certain partial derivatives, such as the advection of a mass-related field, to be evaluated over a smaller grid interval. This increases the spatial resolution while decreasing the effects of truncation error on the solution. Because the grid interval for these partial derivatives is decreased, however, models utilizing staggered grids must utilize a smaller time step to maintain computational stability. Consider the CFL stability criterion in its most generic form. If the grid spacing is halved, the time step must also be halved. However, to achieve the same halving of the grid spacing without grid staggering would require double the number of grid points in both the *x* and *y* directions. The added computational expense from this is greater than that associated with a reduced time step. Further, it can be shown that wave dispersion on certain staggered grids is more realistic than on an unstaggered grid.

Arakawa and Lamb (1977, *Methods of Computational Physics*) define five horizontal model grids, referred to as the Arakawa "A" through "E" grids. These are depicted in Figure 2. The Arakawa "A" grid is an unstaggered grid; all model variables are defined at the same points, whether at the grid corners (as pictured) or at the grid center. All other grids are staggered grids. For the "B" grid, mass-related variables are defined at the grid corners while velocity variables are defined at the grid center. Equivalently, mass-related variables may be defined at the grid center and velocity variables at the grid corners. The NAM model is an example of a model that uses a "B" grid. The Arakawa "E" grid is akin to the Arakawa "B" grid, except rotated 45°. On "E" grids, adjacent points for a given variable are along diagonals rather than a purely horizontal or vertical path. The Hurricane WRF, or HWRF, model is an example of a model that uses an "E" grid.

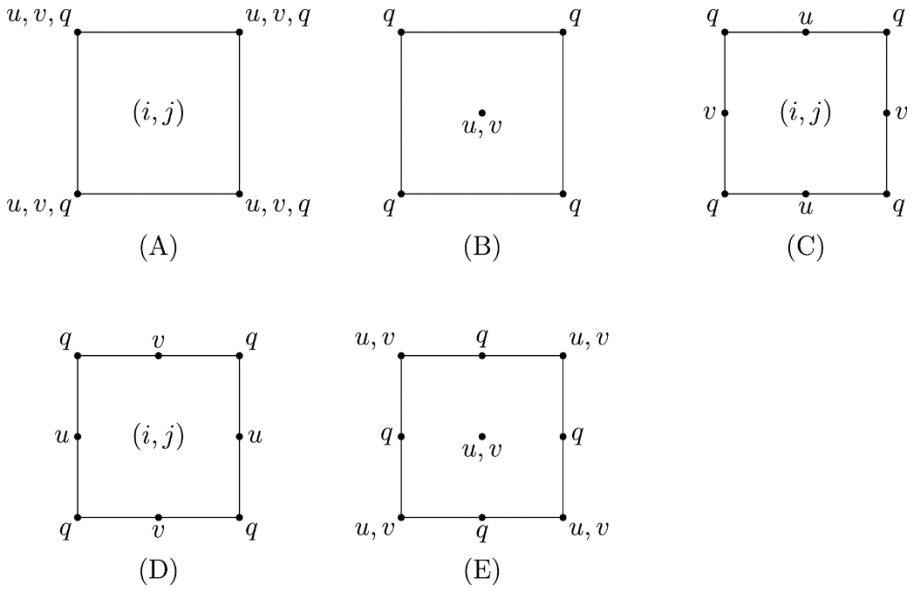


Figure 2. Spatial discretization of model variables on the Arakawa "A" through "E" grids. In each panel, *u* and *v* are the horizontal wind components and *q* is a generic mass-related variable. Please refer to the text for further details regarding each grid. Figure obtained from Wikimedia Commons via author JuliusSimplus and used under a CC BY-SA 3.0 license.

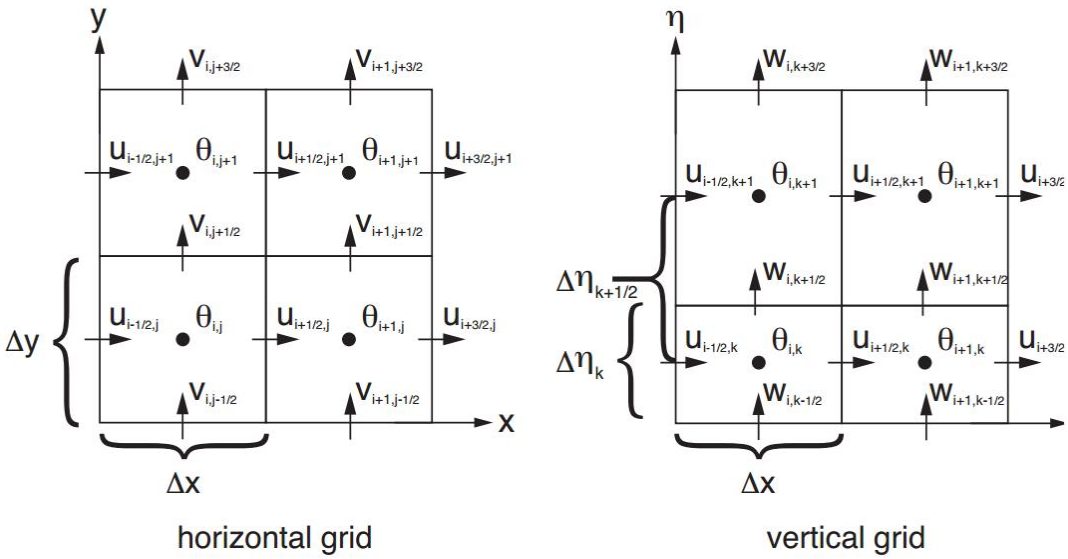


Figure 3. (left) Arakawa "C" grid as utilized within the WRF-ARW model, with horizontal velocities (*u*, *v*) defined normal to grid cell edges and mass-related fields (θ, μ, q_m, p, α) defined at grid centers. (right) Vertical staggered grid as utilized within the WRF-ARW model, with vertical velocity ω and geopotential ϕ defined at the top and bottom of a vertical grid cell. Note that ω and ϕ are defined atop mass-related fields. Figure reproduced from http://www2.mmm.ucar.edu/wrf/users/docs/arw_v3.pdf, their Figure 3.2.

As compared to the "B" grid, velocity variables *u* and *v* on the "C" grid are further staggered with respect to each other. This staggered grid is more commonly depicted in the form given by Figure 3 (left), where mass-related variables are defined at the grid center and *u* and *v* are defined normal to grid cell edges. This configuration to the model kinematic fields enables for the horizontal divergence – which appears in many of the primitive equations – to be defined coincident with the mass-related variables, a desirable trait. Many modern grid point models, including the WRF-ARW and MPAS models, use Arakawa "C" grids.

The "D" grid is a 90° rotation of a "C" grid. Rather than *u* and *v* defined normal to grid cell edges, *u* and *v* on the "D" grid are parallel to grid cell edges. This configuration to the model kinematic fields allows for circulation and vorticity to be defined coincident with the mass-related variables. This does not provide the advantages that horizontal divergence does with the "C" grid, however. Furthermore, studies have shown that the "D" grid does not handle wave dispersion as well as the "B," "C," and "E" grids and, consequently, it is not commonly used in modern models.

The most common type of staggered vertical grid is one where vertical velocity and geopotential are defined at the top and bottom of a vertical grid cell and all other model variables are defined at the center of a vertical grid cell. A graphical example is given in Figure 3 (right). Why would vertical velocity and geopotential be defined on the grid cell edges? If one takes the total derivative of the definition of the geopotential ($\Phi = gz$), a diagnostic relationship between the geopotential and the vertical velocity ($w = \mathrm{D}z/\mathrm{D}t$) results. Plus, defining vertical velocity at the top and bottom of vertical grid cells allows us to specify no vertical velocity through the lower model boundary as a lower model boundary condition, a desirable practice.

How is grid staggering manifest in practice? Consider the simplified form of the *u*-momentum equation introduced earlier in this lecture:

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

Let us assume that model variables are defined on an Arakawa "C" grid and that we are using an Eulerian formulation for representing partial derivatives. As illustrated by Figure 3, the only model variable defined at the grid points where *u* is defined is *u* itself. Arakawa "C" grid staggering of the mass-related variables allows the horizontal pressure gradient term to also be evaluated at the grid points where *u* is defined. However, interpolation of the other model variables to the *u* grid points is necessary to be able to solve the equation above.

For density ρ, a mass-related field, this is straightforward: compute the average of the densities defined at the adjacent mass-related grid points in the *x*-direction (e.g., at points *x+1/2* and *x-1/2*). For *v* and *w*, this is more complicated due to how they are staggered with respect to *u*. First, compute the average *v* or *w* at the mass-related grid points in the *x*-direction adjacent to the *u* grid point. Next, compute the average *v* or *w* between these adjacent mass-related grid points in the *x*-

direction. The interpolation between grids required with staggered grids introduces a potential source of error into the model formulation. However, the potential drawbacks associated with such interpolation are small compared to the benefits of reduced truncation error and more accurate representation of wave dispersion associated with staggered grids and, in particular, the "C" grid.

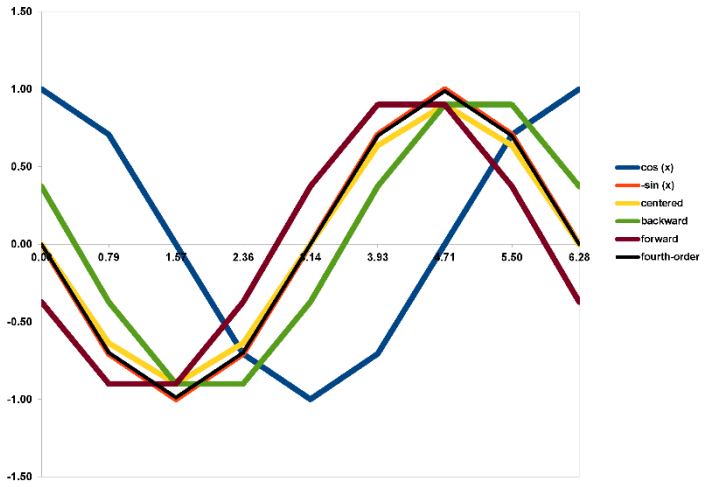
Truncation Error

Conceptually, **truncation error** in space is fairly straightforward: it is the error in approximating a partial derivative of some function *f* with finite differences that results from truncating the Taylor series expansion of *f*. Lower-order-accurate finite difference approximations truncate more terms from the Taylor series expansion; generally speaking, they also use a smaller number of grid points to calculate the finite difference. As a result, lower-order-accurate finite difference approximations are associated with larger truncation error than their higher-order-accurate brethren. Truncation error also exists in time, manifest as variability between individual model time steps that is not accounted for during the model integration.

One can consider truncation error qualitatively. Consider a generic function $f(x) = \cos x$. It has a first partial derivative with respect to x that is exactly equal to $-\sin x$. In Figure 4a, a nine-point grid with grid points every $\pi/4$ radians is used to represent this function (blue); its first derivative with respect to x (orange); and the forward (brown), backward (green), centered (yellow), and fourth-order (black) finite difference approximations to its first derivative with respect to x. As the order of accuracy increases, the match to the true solution improves and truncation error decreases. For this example, the fourth-order finite difference approximation is almost exact, and a sixth-order finite difference approximation is an even better match to the analytic solution.

In Figure 4b, a seventeen-point grid with grid points every $\pi/8$ radians is used to represent the same function as above. Note that the function and its first derivative with respect to x are smoother owing to the decreased grid spacing. The decreased grid spacing also has the benefit of improving each finite difference approximation's match to the analytic solution. Here, both the centered and fourth-order approximations are nearly exact matches to the analytic solution. From this, we can deduce that the truncation error is also a function of the horizontal grid spacing relative to the wavelength of the feature being represented on that grid.

(a)



(b)

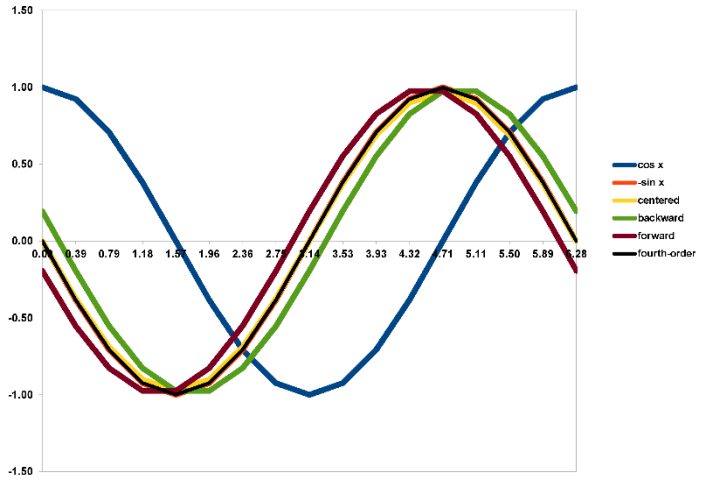


Figure 4. (a) A nine-point grid with grid points every π/4 radians is used to represent *f(x)* = cos *x* (blue); its first derivative (-sin *x*, orange); and the forward (brown), backward (green), centered (yellow), and fourth-order (black) finite difference approximations to its first derivative. (b) As in (a), except on a seventeen-point grid with grid points every π/8 radians.

We can also consider truncation error quantitatively. For instance, let $f(x) = A\cos(kx)$, where A is the amplitude, k is a wavenumber equal to $2\pi/L$, and L is the wavelength. The exact value of the first partial derivative of this function with respect to x is $-Ak\sin(kx)$. A centered finite difference approximation to this derivative, evaluated at a generic point x, is given by:

$$\frac{\Delta f}{\Delta x} = \frac{A\cos\left(k\left(x+\Delta x\right)\right) - A\cos\left(k\left(x-\Delta x\right)\right)}{2\Delta x}$$

The cos terms in the approximation above can be rewritten using trigonometric identities, where $\cos(a+b) = \cos a\cos b - \sin a\sin b$ and $\cos(a-b) = \cos a\cos b + \sin a\sin b$. Doing so, we obtain:

$$\frac{\Delta f}{\Delta x} = \frac{A\left(\left(\cos kx \cos k\Delta x - \sin kx \sin k\Delta x\right) - \left(\cos kx \cos k\Delta x + \sin kx \sin k\Delta x\right)\right)}{2\Delta x} = \frac{-A \sin kx \sin k\Delta x}{\Delta x}$$

One representation for truncation error is given by the ratio of the finite difference approximation to the analytic or exact solution, i.e.,

$$\frac{\Delta f / \Delta x}{\partial f / \partial x}$$

Where this ratio is approximately equal to 1, truncation error is small. Where this ratio departs from 1, truncation error is large.

For this *f(x)* and its centered finite difference approximation, we obtain:

$$\frac{\dfrac{-A \sin kx \sin k\Delta x}{\Delta x}}{-Ak \sin kx} = \frac{\sin k\Delta x}{k\Delta x}$$

Because $k = 2\pi/L$, $k\Delta x \propto \Delta x / L$. Thus, as we stated before, truncation error is a function of the horizontal grid spacing relative to the wavelength of the feature being represented on that grid.

The small angle theorem states that as sin $k\Delta x$ approaches zero, $\sin k\Delta x \approx k\Delta x$. Thus, when Δx is small relative to the wavelength (e.g., a large number of grid points over the wavelength), sin $k\Delta x$ approaches sin (0), which is zero. Thus, for small Δx, $\sin k\Delta x \approx k\Delta x$ and the ratio between the approximate and exact solutions approaches 1. Thus, truncation error in this case is small.

The inverse is true as Δx becomes large relative to the wavelength. The maximum allowable value for Δx is $L/2$ (i.e., a domain with three grid points upon which only a wave with wavelength $2\Delta x$ may be resolved). There, $k\Delta x = 2\pi\Delta x / L = 2\pi L / 2 / L = \pi$, for which sin $k\Delta x$ = sin π = 0. Stated equivalently, *truncation error for short wavelength features is large, whereas it is small for longer wavelength features* (for a given Δx)!

Analogous expressions to the one above can be obtained for any finite differencing scheme. For the forward, backward, and fourth-order finite difference approximations, these take the form:

$$\frac{-\cos k\Delta x}{k\Delta x \tan kx} + \frac{1}{k\Delta x \tan kx} + \frac{\sin k\Delta x}{k\Delta x} \quad \text{(forward)}$$

$$-\frac{1}{k\Delta x \tan kx} + \frac{\cos k\Delta x}{k\Delta x \tan kx} + \frac{\sin k\Delta x}{k\Delta x} \quad \text{(backward)}$$

$$\frac{\sin k\Delta x\left(4 - \cos k\Delta x\right)}{3k\Delta x} \quad \text{(fourth-order)}$$

All three are dependent upon $k\Delta x$, or the horizontal grid spacing relative to the wavelength. The forward and backward approximations are also dependent upon kx, or the position within the wave.

A graph of this ratio for the centered and fourth-order finite difference approximations is provided below in Figure 5. Depicted along the x-axis is wavenumber n, equal here to wavelength L divided by horizontal grid spacing Δx. For smaller values of n, a feature's wavelength is small relative to the model grid spacing and thus has greater truncation error. In this sense, model resolution can be defined relative to the truncation error; e.g., the wavelength of the smallest feature that can be represented on the model grid with a minimum of truncation error (e.g., where the ratio is greater than or equal to ~0.95). For the fourth-order approximation, this appears to be ~8Δx, whereas for the centered approximation, this appears to be ~13Δx.

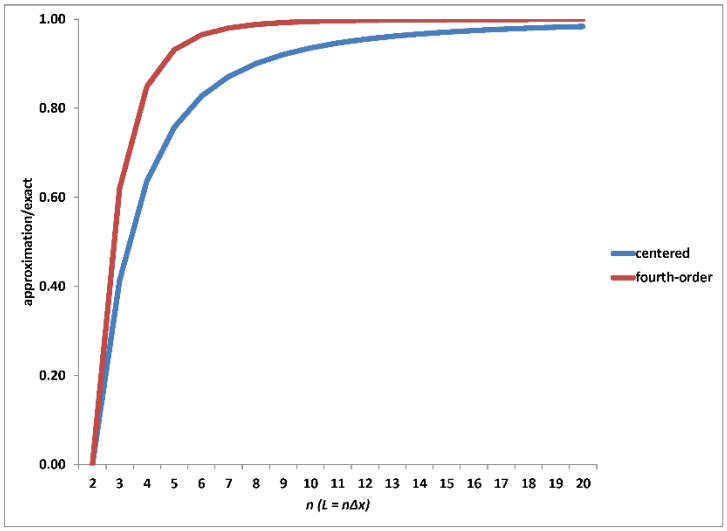


Figure 5. The ratio of the value of the finite difference approximation of the first partial derivative of $f(x) = A \cos kx$ to its exact value, as a function of the number of grid points n used to resolve the wave, for the centered (blue) and fourth-order (red) finite difference approximations. Adapted from Warner (2011), their Figure 3.22.

Furthermore, for any given n, the fourth-order finite difference approximation is associated with less truncation error than the centered approximation. Again, this is particularly evident at smaller wavelengths. Thus, the horizontal grid spacing for a given model simulation must be chosen in light not just of the smallest features desired to be resolved on the model grid but also the order of accuracy used by the finite difference approximations available within the model.

As we will see through the remainder of our discussion of numerical methods, short wavelengths – those below which the model can reasonably resolve, whether viewed in light of truncation error or some other means – pose a particular challenge to model accuracy and stability. Later, we will consider methods for addressing this within numerical model simulations.