

1 Time-Integrators

Francis X. Giraldo,
giraldo@nrlmry.navy.mil,
www.nrlmry.navy.mil/~giraldo/projects/nseam.html

1.1 Introduction

Roughly speaking, there are 2 classes of TIs:

1. Eulerian Methods (fixed-frame - e.g., a rock at the bottom of a flowing river)
2. Lagrangian Methods (moving-frame - e.g., a leaf on the surface of a flowing river)

Within these two classes there are 3 subclasses

1. Fully Explicit Methods
2. Fully Implicit Methods
3. Semi-Implicit Methods

Central to the discussion on time-integrators is the concept of a Courant number. What is a Courant number? The Courant number in one-dimension is defined as

$$C \equiv \frac{u\Delta t}{\Delta x}$$

where u is the characteristic wave speed of the system, Δt is the time-step of the numerical model, and Δx is the spacing of the grid in the numerical model.

Definition 1 *The Courant number is a measure of how much information traverses (u) a computational grid cell (Δx) in a given time-step (Δt). So for example, in an explicit Eulerian method the Courant number cannot be greater than one because a Courant number greater than one means that information is propagating through more than one grid cell at each time step. This means that the time-integrator does not have time to properly interpret what is physically happening which means that the solution will become unstable and the model will blow up!*

Definition 2 *The Courant-Friedrichs-Lewy (CFL) condition is the maximum allowable Courant number that a time-integrator can use. Explicit methods have CFL conditions near 1 but there are some that have much larger CFL conditions. An example is the 4th order Runge-Kutta method.*

1.2 Eulerian Methods

In Eulerian methods the governing equations are discretized in time along an Eulerian (fixed) frame of reference. Therefore, the equations are written in Eulerian form, that is,

$$\frac{\partial q}{\partial t} = S(q) \equiv N(q) + L(q)$$

where $S(q)$ are the source terms and they are split into the nonlinear, $N(q)$, and linear, $L(q)$, terms. Thus in this frame of reference, the advection equation is written as

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

Eulerian methods are the methods taught in numerical analysis classes in undergraduate and most graduate courses. They are the easiest methods to understand and quite easy to code. This does not mean that they are simplistic methods, though. They are quite powerful and have been used for all sorts of applications in the computational sciences.

Example 3 *The most common Eulerian method used in GFD is the explicit leapfrog method. That is, we integrate in time by the following 2nd order finite-difference approximation in time:*

$$\frac{q^{n+1} - q^{n-1}}{2\Delta t} = - \left(u \frac{\partial q}{\partial x} \right)^n$$

1.3 Lagrangian Methods

In contrast, Lagrangian methods discretize the equations along a Lagrangian (moving) frame of reference. Thus the equations are written in Lagrangian form

$$\frac{dq}{dt} = S(q) \equiv N(q) + L(q)$$

where

$$\frac{d}{dt} = \frac{\partial}{\partial t} + \frac{d\mathbf{x}}{dt} \frac{\partial}{\partial \mathbf{x}}$$

and

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}$$

defines the velocity. Thus in this frame of reference, the advection equation,

$$\frac{\partial q}{\partial t} + u \frac{\partial q}{\partial x} = 0$$

is written as

$$\frac{dq}{dt} = 0$$

There are many variants of Lagrangian methods and they include:

1. fully Lagrangian methods,
2. arbitrary Lagrangian-Eulerian (ALE),
3. semi-Lagrangian methods, and
4. operator-integration-factor splitting (OIFS) methods.

1.4 Explicit Methods

A good example of an explicit method is the leapfrog method which is written as

$$q^{n+1} = q^{n-1} + 2\Delta t S(q^n)$$

Note that the only unknown is on the LHS. The RHS only contains known quantities. Discretized in space by a spatial discretization method we would get

$$q_i^{n+1} = q_i^{n-1} + 2\Delta t \widehat{S}_i(q^n)$$

where $i = 1, \dots, N$

Remark 1 *The leapfrog method is perhaps the most common explicit Eulerian method found in geophysical fluid dynamics because it is a symplectic method, that is, a method which preserves all Lagrangian invariants. For hyperbolic equations the ideal method is one which has no numerical diffusion (damping). However, a second order method applied to a first order set of equations such as the hydrostatic primitive equations (HPE) will yield two solutions: a physical solution and a computational mode. The computational mode in leapfrog is as large as the physical solution and through nonlinear interactions can pollute the physical solution and cause instabilities. This is typically fixed through the use of a Robert-Asselin time-filter. However, this then reduces the order of accuracy to first order.*

Examples of explicit methods are:

1. Euler's method (first order) , which can be written as

$$q^{n+1} = q^n + \Delta t S(q^n)$$

and is a one-step method

2. the Runge-Kutta methods (second order), which can be written as

$$\begin{aligned} q^{n+\frac{1}{2}} &= q^n + \frac{\Delta t}{2} S(q^n) \\ q^{n+1} &= q^n + \Delta t S(q^{n+\frac{1}{2}}) \end{aligned}$$

and are one-step multi-stage methods

3. and Adams-Bashforth methods (second order), which can be written as

$$q^{n+1} = q^n + \frac{\Delta t}{2} [3S(q^n) - S(q^{n-1})]$$

and are one-stage multi-step methods.

Note, however, that there is a plethora of RK and AB methods. We only mention these second order methods for the sake of brevity.

The only disadvantage of explicit methods is that they impose stringent time-step restrictions of the type

$$C \equiv \frac{u\Delta t}{\Delta x} < 1$$

where $u = U + \sqrt{\phi}$ is the characteristic velocity of the hydrostatic primitive equations. U is the wind velocity and ϕ is the geopotential height and the square root represents the gravity wave speed. The gravity wave speeds are far larger than the wind speeds $\sqrt{\phi} \gg U$.

Remark 2 *For the hydrostatic primitive equations, the gravity waves may be an order of magnitude larger than the Rossby waves. However, gravity waves contain very little energy and so for short to medium-range weather forecasting the proper representation of gravity waves is not very critical. Typically, in NWP the strategy is to use implicit methods for the gravity waves and explicit or Lagrangian methods for the Rossby waves.*

1.5 Implicit Methods

There are a variety of implicit methods to choose from. Time-integrators are highly dependent on the type of equations which you are solving. For the purposes of this discussion we shall only address hyperbolic equations such as the wave equation. For this type of equation the most common implicit time-integrators are:

1. Crank-Nicholson (a.k.a. trapezoidal rule) which is written as

$$q^{n+1} = q^n + \frac{\Delta t}{2} [S(q^{n+1}) + S(q^n)]$$

and is $O(\Delta t^2)$ accurate.

2. Backward Difference Formulas (BDF) which are written as

$$q^{n+1} = \sum_{i=0}^M \alpha_i q^{n-i} + \gamma \Delta t S(q^{n+1})$$

and are $O(\Delta t^{M+1})$ accurate.

The simplest example of an implicit method is the backward Euler's method which is written as

$$\begin{aligned}
q^{n+1} &= q^n + \Delta t S(q^{n+1}) \\
&= q^n + \Delta t N(q^{n+1}) + \Delta t L(q^{n+1})
\end{aligned}$$

Discretizing in space and rearranging such that the unknowns are on the LHS yields

$$q_i^{n+1} - \Delta t \widehat{N}_i(q^{n+1}) - \Delta t \widehat{L}_i(q^{n+1}) = q_i^n$$

Thus not only will we have to solve a *matrix problem* but the matrix problem will be a *nonlinear* one. What do we do? First, we linearize the problem using Newton's method with the problem statement

$$F^{m+1} = F^m + \frac{\partial F^m}{\partial q}(q^{m+1} - q^m) = 0$$

for $m = 0, \dots, M$ where

$$F = q_i^{n+1} - \Delta t \widehat{N}_i(q^{n+1}) - \Delta t \widehat{L}_i(q^{n+1}) - q_i^n \equiv 0$$

This is expensive because we will require a few m iterations to converge to the nonlinear problem but at each m iteration the linear matrix problem

$$\frac{\partial F^m}{\partial q}(q^{m+1} - q^m) = -F^m$$

needs to be solved which will also require quite a few iterations (if using an iterative solver, which you most likely will). There is an emerging method called Jacobian-free Newton-Krylov (JFNK) methods but these methods are extremely complicated and the success of this method is determined by how good your Krylov method is. Examples of Krylov methods are:

1. the Generalized Minimum Residual Method (GMRES) and
2. the conjugate gradient squared (CGS).

Even if we were able to solve this problem efficiently with an implicit Eulerian method there will still be a limit on the maximum allowable time-step. Clearly, an implicit Eulerian method will not have a CFL condition, but due to the order of accuracy in time we cannot just pick extremely large time-steps and get good accuracy. Thus for implicit Eulerian methods the time-step restriction is one of accuracy rather than stability.

Is there another option?

1.6 Semi-Implicit Methods

The semi-implicit method splits the source terms into linear and nonlinear terms. Therefore, we can then discretize the nonlinear terms explicitly and the linear terms implicitly.

A good example of a semi-implicit method is the leapfrog with Crank-Nicholson

$$q^{n+1} = q^{n-1} + 2\Delta t N(q^n) + 2\Delta t \left[\frac{1}{2}L(q^{n+1}) + \frac{1}{2}L(q^{n-1}) \right]$$

Discretizing in space and rearranging such that the unknowns are on the LHS yields

$$q_i^{n+1} - \Delta t \widehat{L}_i(q^{n+1}) = q_i^{n-1} + 2\Delta t \widehat{N}_i(q^n) + \Delta t \widehat{L}_i(q^{n-1})$$

Thus we still have to solve a matrix problem but at least it is a linear one. If the wind velocity, U , is contained in N and the geopotential, ϕ , in L then the CFL condition is

$$C \equiv \frac{U\Delta t}{\Delta x} < 1$$

which is much less stringent than the explicit restriction

$$C \equiv \frac{(U + \sqrt{\phi}) \Delta t}{\Delta x} < 1$$

At T239 L30 the semi-implicit time-step is *10 times larger* than the explicit time-step! However, a CFL condition still remains so how can we make it go away?

1.7 Implicit Lagrangian Methods

Typically, the nonlinearities in the equations come from the advection terms

$$\mathbf{u} \cdot \nabla q$$

So that if we were to write the governing equations in Lagrangian form then we would get something like this

$$\frac{dq}{dt} = S(q) \equiv L(q)$$

Discretizing this equation by the leapfrog with Crank-Nicholson yields

$$q^{n+1} = \widehat{q}^{n-1} + \Delta t [L(q^{n+1}) + L(\widehat{q}^{n-1})]$$

Discretizing in space and rearranging gives

$$q_i^{n+1} - \Delta t \widehat{L}_i(q^{n+1}) = \tilde{q}_i^{n-1} + \Delta t \widehat{L}_i(\tilde{q}^{n-1})$$

which again requires solving a linear matrix system but there is no longer any CFL condition. You can now take Δt as large as you want (the only restriction is something called the Lipschitz condition and the condition number of the resulting matrix). However, we still have not addressed how the departure point values, \tilde{q}^{n-1} , are obtained.

Definition 4 *The Lipschitz condition is a measure of the smoothness of the solution. For example, assuming that there is a discontinuity somewhere in the flow. A true discontinuity (such as a shock wave) is multi-valued and so one cannot use information along the characteristics (i.e., departure point values) that begin at this shock because the solution is multi-valued there.*

1.7.1 Departure Point Values

Assuming a constant wind velocity we can discretize the trajectory equation

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}$$

as follows

$$\tilde{\mathbf{x}}^{n-1} = \mathbf{x}^{n+1} - 2\Delta t \mathbf{u}$$

which is the leapfrog method. In most applications high-order methods such as Runge-Kutta 2nd, 3rd, or 4th order are used. Now, the departure point value is then given as

$$\tilde{q}^{n-1} = q(\tilde{\mathbf{x}}^{n-1}, t^{n-1})$$

The only difficulty is how to actually compute this departure point value.

Typically, $\tilde{\mathbf{x}}^{n-1}$ will not lie on a grid point and so some interpolation is required. This is the crux of the problem. How to obtain high-order interpolation in order to represent the departure point value accurately without incurring too high a computational cost? Let's not worry about that for now but instead concentrate on how to apply this to the 1D advection equation.

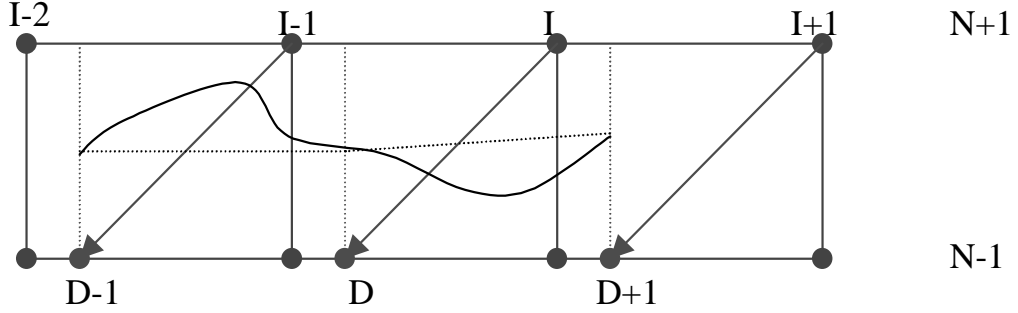


Figure 1: Graphical representation of the SL method.

1.7.2 Application to the 1D Advection Equation

In Eulerian form the 1d advection equation is

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

and in Lagrangian form

$$\frac{dq}{dt} = 0$$

with

$$\frac{dx}{dt} = u.$$

Discretizing in time along the characteristics

$$q^{n+1} = \hat{q}^{n-1}$$

where the location of the departure point is obtained from

$$x^{n-1} = x^{n+1} - 2\Delta t u.$$

Integrating the weak integral form,

$$\int_{\Omega} \psi q^{n+1} d\Omega = \int_{\Omega} \psi \hat{q}^{n-1} d\Omega$$

assuming linear elements, and letting

$$q(x) = \sum_{j=1}^2 \psi_j(x) q_j$$

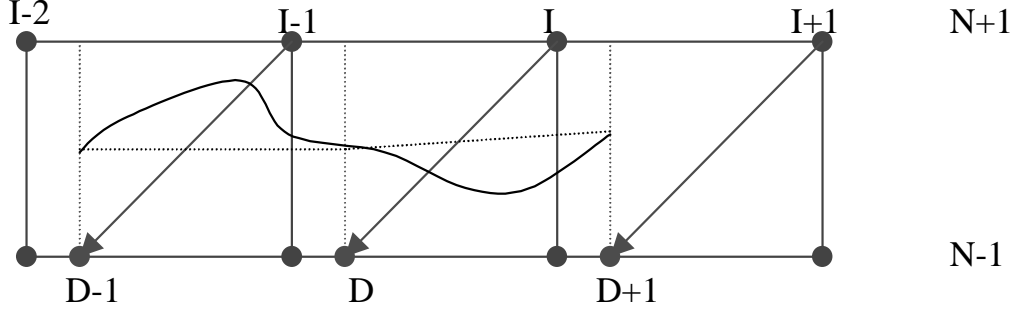


Figure 2: Graphical representation of the SL method.

we get

$$\int_{\Omega^{n+1}} \psi_i \psi_j d\Omega^{n+1} q_j^{n+1} = \int_{\Omega^{n+1}} \psi_i \psi_j d\Omega^{n+1} \tilde{q}_j^{n-1}.$$

Integrating yields

$$\frac{\Delta x}{6} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix}^{n+1} = \frac{\Delta x}{6} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} \tilde{q}_1 \\ \tilde{q}_2 \end{pmatrix}^{n-1}.$$

The contribution to the gridpoint i is

$$\frac{\Delta x}{6} [q_{i-1} + 4q_i + q_{i+1}]^{n+1} = \frac{\Delta x}{6} [\tilde{q}_{i-1} + 4\tilde{q}_i + \tilde{q}_{i+1}]^{n-1}.$$

Now, let us further assume that the Courant number is less than one, that is that the fluid particles travel less than one element per time step. This is depicted in figure 2 where the departure point of the gridpoint i (d) is between the gridpoints $i - 1$ and i .

We can now interpolate the departure point values as follows

$$\tilde{q}_{i-1}^{n-1} \equiv q_{d-1}^{n-1} = Cq_{i-2}^{n-1} + (1 - C)q_{i-1}^{n-1}$$

$$\tilde{q}_i^{n-1} \equiv q_d^{n-1} = Cq_{i-1}^{n-1} + (1 - C)q_i^{n-1}$$

$$\tilde{q}_{i+1}^{n-1} \equiv q_{d+1}^{n-1} = Cq_i^{n-1} + (1 - C)q_{i+1}^{n-1}$$

where

$$C = \frac{u\Delta t}{\Delta x}$$

is the Courant number.

Substituting yields

$$\begin{aligned} \frac{\Delta x}{6} [q_{i-1} + 4q_i + q_{i+1}]^{n+1} = \\ \frac{\Delta x}{6} [Cq_{i-2} + (1 + 3C)q_{i-1} + (4 - 3C)q_i + (1 - C)q_{i+1}]^{n-1} \end{aligned}$$

which can be written as

$$\frac{\Delta x}{6} \sum_{k=i-1}^{i+1} \alpha_k q_k^{n+1} = \frac{\Delta x}{6} \sum_{k=i-2}^{i+1} \beta_k q_k^{n-1}$$

where

$$\beta_k = \beta_k(C).$$

Clearly, if we desired higher order accuracy we would have to select a higher order interpolation scheme for the departure point value. For $N = 4$ we would get

$$\frac{\Delta x}{6} \sum_{k=i-4}^{i+4} \alpha_k \varphi_k^{n+1} = \frac{\Delta x}{6} \sum_{k=i-5}^{i+4} \beta_k \varphi_k^{n-1}$$

where

$$\beta_k = \beta_k(C^4)$$

Thus, the SL method obtains its order of accuracy through the order of interpolation at the departure points.

2 Space-Time Combinations

2.1 Accuracy

- There have been misunderstandings regarding Lagrangian methods; one misunderstanding states that it is sufficient to use linear interpolation for the velocity field and a 2nd order scheme for the trajectory equation (ECMWF, Canadian Regional Model, etc.)

2.1.1 Spectral Element Semi-Lagrangian (SESL) Method

- In Falcone and Ferretti (1998, SIAM J. Num. Analysis) it was shown that the error for Lagrangian methods is of the order

$$O\left(\Delta t^k + \frac{\Delta x^{N+1}}{\Delta t}\right) \quad (1)$$

where k, N represent the order of the trajectory approximation and the order of the interpolation functions, respectively. In other words for the trajectory equation

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}$$

k represents the order of accuracy of the numerical scheme (e.g., Runge-Kutta) used to solve this equation while N represents the order of interpolation used to compute the variables at the departure points

- Therefore, if we use either high order k with low order N or vice-versa, then we limit the overall accuracy of the scheme
- In all previous SLM implementations researchers used low order k (≤ 2) and N (≤ 3)!
- However, no one tried to increase both values! It may have been due in part to naivete but it mainly had to do with the fact that these high order local methods (i.e., *spectral element methods*) simply did not exist
- In order to show the kinds of results attainable with this method, let us look at the L_2 error for the advection equation on the sphere
- In Giraldo (1998, Journal of Computational Physics), we developed a new SL method (SESL=spectral element semi-Lagrangian).
 - In previous semi-Lagrangian methods the time and space discretizations were separate operations. For example, a 2nd order trajectory equation is solved with, at most, third order interpolation with the spectral method (ECMWF).

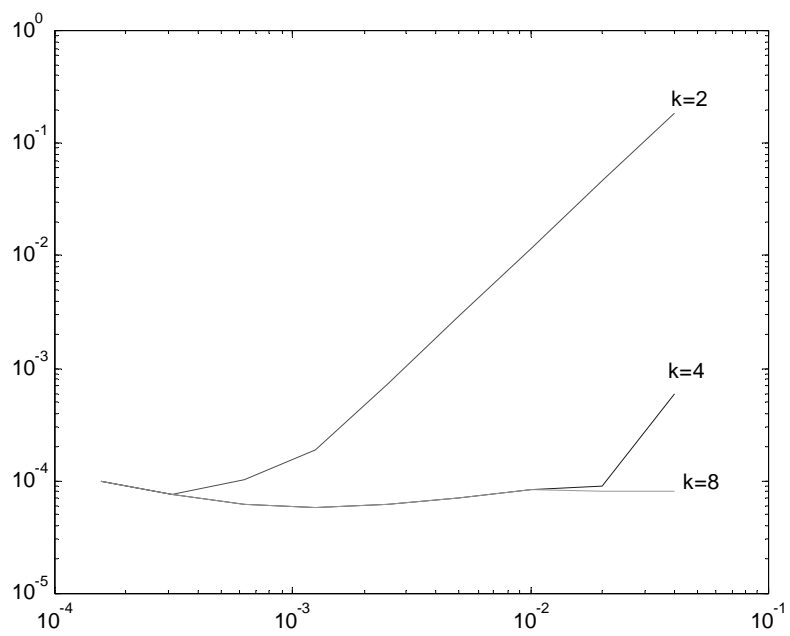


Figure 3: SESL: The L_2 error as a function of time step in days for the spherical advection equation for $N = 8$.

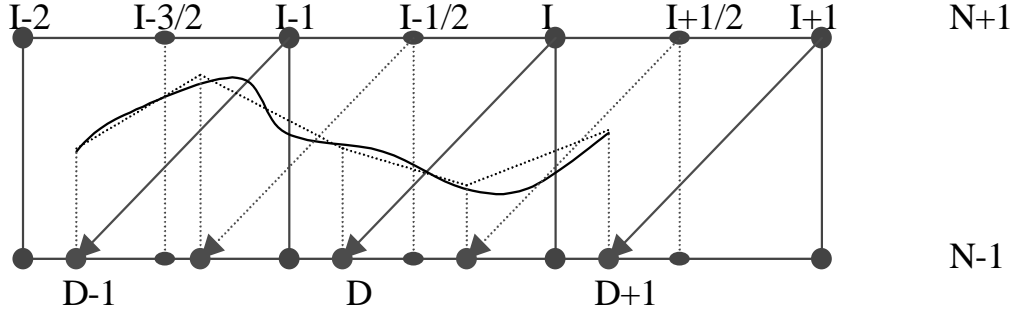


Figure 4: Graphical representation of the SESL method ($N = 2$).

- In SESL, the interpolation stencils required by the SL are the same basis functions used in the SE. Therefore if we are using $N = 8$ order basis functions, then we are also using 8th order interpolation in the SL. This method was shown to have superconvergence characteristics and better filtering capabilities due to the admittance of flux-limiting.
 - For $N = 2$ we get the following stencil
- Looking at the Falcone error analysis we see that the optimal time step for a given grid resolution is the one that balances the time and spatial discretization errors. Following Malevsky [?] we write the time step as

$$\Delta t = \Delta x^\alpha$$

- Equating the first and second terms in (1) yields the optimal time-step

$$\Delta t = \Delta x^{\frac{N+1}{k+1}}$$