Hyperbolic PDEs and Finite-Volume Methods IV

Ammar H. Hakim (`ammar@princeton.edu`) [1]

[1]Princeton Plasma Physics Laboratory, Princeton, NJ

Princeton University, Course AST560, Spring 2021

Steps in constructing finite-volume method

$$\frac{\partial \mathbf{Q}_j}{\partial t} + \frac{\mathbf{G}(\mathbf{Q}^+_{j+1/2}, \mathbf{Q}^-_{j+1/2}) - \mathbf{G}(\mathbf{Q}^+_{j-1/2}, \mathbf{Q}^-_{j-1/2})}{\Delta x} = 0$$

To completely specify a finite-volume scheme we must design algorithms for each of the following three steps:

- **Step 1**: A **recovery scheme** (possibly with limiters) to compute the left/right interface values $\mathbf{Q}^\pm$ at each interface using a set of cell-average values around that interface,
- **Step 2**: A **numerical flux function** that takes the left/right values and returns a consistent approximation to the physical flux, and
- **Step 3**: A **time-stepping scheme** to advance the solution in time and compute the cell-averages at the next time-step.

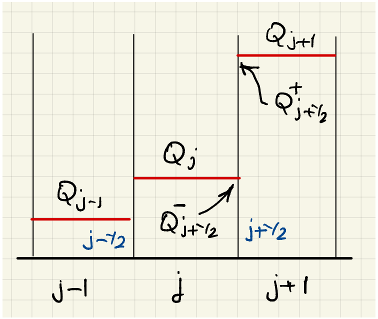
Essence of the finite-volume method

Instead of computing one edge value we will compute *two* values: one the left and one on right of cell-edge. We will next define a *numerical flux function*

$$\mathbf{G} = \mathbf{G}(\mathbf{Q}^{-}_{j+1/2}, \mathbf{Q}^{+}_{j+1/2})$$

with *consistency* condition

$$\lim_{\mathbf{Q}_{L,R}\rightarrow Q} \mathbf{G}(\mathbf{Q}_L, \mathbf{Q}_R) = \mathbf{F}(\mathbf{Q})$$



In terms of the numerical flux function the FV update formula becomes

$$\frac{\partial \mathbf{Q}_j}{\partial t} + \frac{\mathbf{G}(\mathbf{Q}^{+}_{j+1/2}, \mathbf{Q}^{-}_{j+1/2}) - \mathbf{G}(\mathbf{Q}^{+}_{j-1/2}, \mathbf{Q}^{-}_{j-1/2})}{\Delta x} = 0$$

Numerical Flux Function

The numerical flux function computes a *consistent* flux at the cell-edge from the cell averages.

$$\lim_{\mathbf{Q}_{L,R}\to Q} \mathbf{G}(\mathbf{Q}_L, \mathbf{Q}_R) = \mathbf{F}(\mathbf{Q}).$$

Examples

- *Central Flux* in which we simply average the flux from the two states at the interface

$$\mathbf{G}(\mathbf{Q}_L, \mathbf{Q}_R) = \frac{1}{2}\left(\mathbf{F}(\mathbf{Q}_L) + \mathbf{F}(\mathbf{Q}_R)\right).$$

- *Upwind Flux* in which we choose the edge on the "upwind" side to account for direction of information flow:

$$\mathbf{G}(\mathbf{Q}_L, \mathbf{Q}_R) = \mathbf{F}(\mathbf{Q}_L)$$

  if information is flowing from left-to-right, and

$$\mathbf{G}(\mathbf{Q}_L, \mathbf{Q}_R) = \mathbf{F}(\mathbf{Q}_R)$$

  if information is flowing from right-to-left. Begs the question: how to determine which direction information is flowing in? Answer: the eigensystem of the hyperbolic equation contains this!

Numerical Flux Function: Lax flux

- A good choice of the numerical flux function is the *local Lax* flux:

$$\mathbf{G}(\mathbf{Q}_L, \mathbf{Q}_R) = \frac{1}{2}\left(\mathbf{F}(\mathbf{Q}_L) + \mathbf{F}(\mathbf{Q}_R)\right) - \frac{|\lambda|}{2}(\mathbf{Q}_R - \mathbf{Q}_L)$$

  where $|\lambda|$ is an estimate of the (absolute) maximum of all eigenvalues at the interface.

- For advection equation this becomes

$$G(f_L, f_R) = \frac{1}{2}a(f_L + f_R) - \frac{|a|}{2}(f_R - f_L)$$

  This works for either sign of advection speed a, automatically giving upwinding.

- Note $|\lambda|$ is only a local (to the interface) *estimate*. You can use a global estimate too: orginal formulation by Peter Lax ("Lax fluxes").

Numerical Flux Function: Systems of equations

- Lax flux is a good "first" flux to use. However, notice it only takes into account a *single* piece of information: maximum eigenvalue.
- For a *linear system* of equations (Maxwell equation) or *locally linearized* nonlinear system we can instead do

$$G(Q_R, Q_L) = \frac{1}{2}\left(F(Q_R) + F(Q_L)\right) - \frac{1}{2}\left(A^+\Delta Q_{R,L} - A^-\Delta Q_{R,L}\right)$$

  where the *fluctuations* $A^\pm \Delta Q$ are defined as

$$A^\pm \Delta Q_{R,L} \equiv \sum_p r^p \lambda_p^\pm (w_R^p - w_L^p) = \sum_p r^p \lambda_p^\pm l^p (Q_R - Q_L).$$

  where $\lambda_p^+ = \max(\lambda_p, 0)$ and $\lambda_p^- = \min(\lambda_p, 0)$.
- Additional care is needed for nonlinear equations like Euler or ideal MHD equations. More on this on Thursday.

Some notation for use in recovery stencils

Example: symmetric recovery across two cells can be written as

$$Q_{i+1/2} = \frac{1}{2}(Q_{i+1} + Q_j) = \frac{1}{2}(d_p + d_m)Q_{i+1/2}$$

Example: central difference scheme for second derivative:

$$\frac{\partial^2 Q_i}{\partial x^2} = \frac{1}{\Delta x^2}(Q_{i+1} - 2Q_j + Q_{i-1}) = \frac{1}{\Delta x^2}(\Delta_p - 2I + \Delta_m)Q_i$$

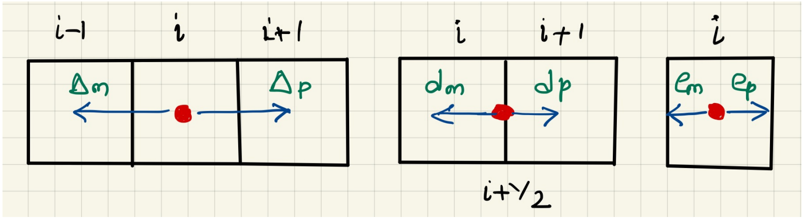
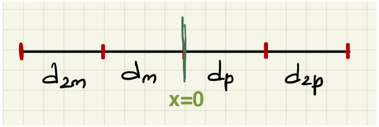


Figure: Basic indexing operators to move from cell to cell, face to cell and cell to face.

Recovery scheme: four-cell stencil, centered scheme



- To construct a four-cell symmetric stencil recovery across an interface we will use a four-cell stencil: $\{d_{2m}, d_m, d_p, d_{2p}\}$
- Setup a local coordinate system with $x = 0$ at the interface and assume a polynomial recovery

$$p(x) = p_0 + p_1 x + p_2 x^2 + p_3 x^3$$

- Match the cell-averages of $p(x)$ in each of the cells $\{d_{2m}, d_m, d_p, d_{2p}\}$ to get a system of linear equations. Solve this system to determine p_0, p_1, p_2, p_3.

Recovery scheme: four-cell stencil, centered scheme

Solving the system of four equations for the four coefficients p_i, $i = 0, \ldots, 3$ yields:

$$
\begin{aligned}
p_0 &= \frac{1}{12}(-d_{2m} + 7d_m + 7d_p - d_{2p})Q \\
p_1 &= \frac{1}{12\Delta x}(d_{2m} - 15d_m + 15d_p - d_{2p})Q \\
p_2 &= \frac{1}{4\Delta x^2}(d_{2m} - d_m - d_p + d_{2p})Q \\
p_4 &= \frac{1}{6\Delta x^3}(d_{2m} - 3d_m + 3d_p - d_{2p})Q.
\end{aligned}
$$

- Notice: stencils of the even coefficients are *symmetric* and the odd coefficients are *anti-symmetric*.
- To compute the interface value we do not really need all of these coefficients but only need to evaluate the recovery polynomial at $x = 0$, i.e we only need $p(0) = p_0$

Recovery scheme: four-cell stencil, centered scheme

To compute the interface value we do not really need all of these coefficients but only need to evaluate the recovery polynomial at $x = 0$, i.e we only need $p(0) = p_0$. Hence, the interface value can be computed from

$$Q^+ = Q^- = \frac{1}{12}(-d_{2m} + 7d_m + 7d_p - d_{2p})Q.$$

Note that due the symmetric nature of the stencil we have only a *single* value at the interface. This means that the numerical flux function at an interface is simply

$$G(Q, Q) = F(Q)$$

from consistency requirements.This completes the spatial finite-volume discretization! The scheme one gets from this is very accurate (even "structure preserving" for Maxwell equations), though not very robust in presence of sharp gradients. (No Free Lunch)

How accurate is any given scheme?

To fix ideas consider we wish to solve the advection equation

$$\frac{\partial f}{\partial t} + \frac{\partial f}{\partial x} = 0$$

Using the four-cell symmetric recovery scheme to compute interface values in the FV update formula we get the semi-discrete scheme *five-cell stencil* update formula:

$$\frac{\partial f_j}{\partial t} = -\frac{1}{\Delta x}\int_{x_{j-1/2}}^{x_{j+1//2}} \frac{\partial f}{\partial x}\, dx = -\frac{1}{12\Delta x}(f_{j-2} - 8f_{j-1} + 8f_{j+1} - f_{j+2})$$

How accurate is this scheme, or what is its order of convergence?

How accurate is any given scheme? Use Taylor series

- Take a Taylor series polynomial around the cell center of cell $I_j = [-\Delta x/2, \Delta x/2]$ locally at $x = 0$

$$T(x) = \sum_{n=0} \frac{T_n}{n!} x^n.$$

- Compute the cell average of this polynomial in each of the stencil cells $\{\Delta_{2m}, \Delta_m, \Delta_p, \Delta_{2p}\}$
- Substitute these averages in the update formula to compute the mean value of the flux gradient in the cell $I_j = [-\Delta x/2, \Delta x/2]$

$$\frac{1}{12\Delta x}(\Delta_{2m} - 8\Delta_m + 8\Delta_p - \Delta_{2p})T = T_1 + \frac{\Delta x^2}{24} T_3 - \frac{21\Delta x^4}{640} T_5 + \ldots$$

- Subtract the exact cell average of the gradient of the Taylor polynomial in cell $I_j = [-\Delta x/2, \Delta x/2]$, i.e.

$$\frac{1}{\Delta x}\int_{-\Delta x/2}^{\Delta x/2} \frac{\partial T}{\partial x}\, dx = T_1 + \frac{\Delta x^2}{24} T_3 + \frac{\Delta x^4}{1920} T_5 + \ldots$$

from the stencil computed value. The remainder term is the error of the scheme.

Symmetric four-cell recovery scheme is fourth-order accurate

The above procedure (needs use of a compute algebra system to simplify the computations) shows that the symmetric four-cell recovery scheme has error that goes like

$$\frac{\Delta x^4}{30} T_5 + O(\Delta x^6)$$

showing the scheme converges with *fourth-order* accuracy $O(\Delta x^4)$ for linear advection equation. (Reducing Δx by 2 reduces error by a factor of 16).

Accuracy is not everything: dispersion and diffusion

- High-order symmetric schemes like the one we derived are very accurate (even "structure preserving" for some problems) but not robust.
- Two other properties of the scheme are important to understand: *dispersion* and *diffusion*. For this we wil derive a *numerical dispersion relation* analogous to dispersion relation we derived for linearized systems.
- Consider a single mode $f(x) = e^{ikx}$ where k is the wavenumber. Compute the cell-average of the mode on each of the cells in the stencil, plug into the stencil formula to derive the *numerical dispersion relation*

$$i\bar{k} = \sum_{m=-N}^{M} c_m e^{ikm\Delta x}$$

where we have written the stencil in the generic form

$$\frac{1}{\Delta x}\sum_{m=-N}^{M} c_m f_{j+m}$$

Symmetric four-cell recovery scheme has no diffusion!

- Note that the numerical dispersion relation will in general give a *complex* effective wavenumber $\overline{k}$.
- The dispersion relation for a hyperbolic equation is $\omega = \lambda k$. Hence, the *real part* of $\overline{k}$ represents dispersion and *imaginary part* of $\overline{k}$ represents diffusion/growth. Obviously, we want imaginary part to be *negative* to avoid solution blow-up!
- The four-cell symmetric stencil has *no imaginary part* of $\overline{k}$. This related to the fact that it is *symmetric* (anti-symmetric stencil coefficients). This is not necessarily a good thing!

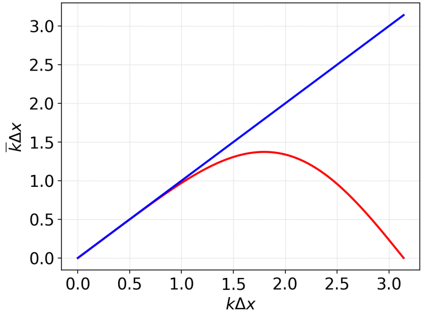


Figure: Real-part of numerical dispersion relation for four-cell recovery scheme. Notice the strong dispersion for higher-k modes

Closer look at numerical dispersion relation

The numerical dispersion relation determines the wave-vector that the *time-propagator* of the scheme sees. Consider:

$$i\bar{k} = i\bar{k}(k) = \sum_{m=-N}^{M} c_m e^{ikm\Delta x}$$

If we are solving a linear advection equation with a single mode solution like $e^{-i\omega_k t}e^{ikx}$, where $\omega_k = \lambda k$ then the effective mode in the discrete scheme will be

$$\bar{\omega}_k = \lambda\bar{k}(k).$$

Hence, the numerical scheme adds *numerical dispersion* to propagating waves as now the phase- and group-velocity are no longer constant.

Godunov's Theorem

- A very important theorem proved by Godunov is that there is **no** *linear scheme* that is "monotonicity preserving" (no new maxima/minima created) and **higher than first-order accurate**!
- Consider a general scheme for advection equation

$$f_j^{n+1} = \sum_k c_k f_{j+k}^n.$$

The discrete slope then is

$$f_{j+1}^{n+1} - f_j^{n+1} = \sum_k c_k \left(f_{j+k+1}^n - f_{j+k}^n \right).$$

Assume that all $f_{j+1}^n - f_j^n > 0$. To maintain monotonicity at next time-step hence one must have all $c_k \geq 0$.

Godunov's Theorem

- First order upwind scheme:

$$f_j^{n+1} = f_j^n - \frac{\Delta t}{\Delta x}(f_j^n - f_{j-1}^n)$$

  this satisfies monotonicity as long as $\Delta t/\Delta x \leq 1$.

- Second order symmetric scheme

$$f_j^{n+1} = f_j^n - \frac{\Delta t}{2\Delta x}(f_{j+1}^n - f_{j-1}^n)$$

  clearly this does not satisfy the condition of monotonicity.

- In general condition on Taylor series to ensure atleast second-order accuracy shows that at least *one* of the c_ks must be negative. Hence, by contradiction, *no such scheme exists*!

Godunov's Theorem: Unfortunate Consequences and Workarounds

- Godunov's Theorem is highly distressing: accurate discretization seems to preclude a scheme free from monotonicity violations
- One way around is to start with a linear scheme that is very accurate and then add some local diffusion to it to control the monotonicity.
- However, Godunov's theorem shows that this "diffusion" must be dependent on the local solution itself and can't be fixed *a priori*. This means a **monotonicity preserving scheme must be nonlinear**, even for linear hyperbolic equations.
- Leads to the concept of *nonlinear limiters* that control the monotonicity violations (adding diffusion to high-k modes). No free lunch: limiters must diffuse high-k modes but this will inevitably lead to issues like inability to capture, for example, high-k turbulence spectra correctly without huge grids.
- Major research project: interaction of shocks, boundary layers and turbulence in high-Reynolds number flows.