Unit 3. Introduction to Computational Fluid Dynamics (CFD)

Lecture 313: Accuracy, conservation and boundary conditions of finite-volume methods

Reference book:

“Introduction to Computational Astrophysical Hydrodynamics” by Zingale.
http://bender.astro.sunysb.edu/hydro_by_example/CompHydroTutorial.pdf

W. Banda-Barragán, 2023

Introduction to CFD: accuracy and stencils

3.3.1 Differences and order of accuracy

In practice, when computing derivatives in a finite-volume discretization, we can use the second-order centered difference from § 1.2.2 treating the finite-volume data as living at the cell-centers and still be second-order accurate. For higher accuracy, we can fit a *conservative interpolant* (as illustrated in exercise 3.2) to a collection of points and then differentiate the interpolant itself.

Notice that the righthand side of all derivative approximations involve some linear combinations of f_i's. We call this the *stencil*. The *width* of the stencil is a measure of how many zones on either side of zone i we reach when computing our approximation.

For example, a second derivative can be discretized as:

$$\left.\frac{d^2 f}{dx^2}\right|_i = \frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2} \tag{3.10}$$

The stencil on the righthand side encompasses 3 zones.

3.3.2 Conservation

The finite-volume grid is useful when dealing with conservation laws. Consider the following system:

$$\frac{\partial \boldsymbol{\mathcal{U}}}{\partial t} + \nabla \cdot \mathbf{F}(\boldsymbol{\mathcal{U}}) = 0 \tag{3.11}$$

where $\boldsymbol{\mathcal{U}}$ is a vector of conserved quantities and $\mathbf{F}(\boldsymbol{\mathcal{U}})$ is the flux of each quantity. This type of system arises, for example, in fluid flow, where the system will represent conservation of mass, momentum, and energy.

Introduction to CFD: stencils and conservation

Consider one-dimension. Integrating this system over a zone, and normalizing by Δx, we have:

$$\frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{\partial \boldsymbol{\mathcal{U}}}{\partial t} dx = -\frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} \frac{\partial \mathbf{F}}{\partial x} dx \tag{3.12}$$

On the left, we can take the time derivative out of the integral, and we are left with the definition of a zone average, so this becomes simply $\partial \langle \boldsymbol{\mathcal{U}} \rangle_i / \partial t$. On the right, we apply the divergence theorem, giving:

$$\frac{\partial \langle \boldsymbol{\mathcal{U}} \rangle_i}{\partial t} = -\frac{1}{\Delta x}\left\{ \mathbf{F}(\boldsymbol{\mathcal{U}})|_{x_{i+1/2}} - \mathbf{F}(\boldsymbol{\mathcal{U}})|_{x_{i-1/2}} \right\} \tag{3.13}$$

Introduction to CFD: stencils and conservation

$$\frac{\partial \langle \mathcal{U} \rangle_i}{\partial t} = -\frac{1}{\Delta x} \left\{ \mathbf{F}(\mathcal{U})|_{x_{i+1/2}} - \mathbf{F}(\mathcal{U})|_{x_{i-1/2}} \right\} \tag{3.13}$$

Independent of how we discretize in time, notice that we have the cell-average on the left and that the righthand side is simply a difference of fluxes through the interfaces of the zone. For the $i+1$ zone, the update would be:

$$\frac{\partial \langle \mathcal{U} \rangle_{i+1}}{\partial t} = -\frac{1}{\Delta x} \left\{ \mathbf{F}(\mathcal{U})|_{x_{i+3/2}} - \mathbf{F}(\mathcal{U})|_{x_{i+1/2}} \right\} \tag{3.14}$$

Notice that this shares the flux at the $x_{i+1/2}$ interface, but with the opposite sign. When all of the updates are done, the flux through each boundary adds to one zone and subtracts from its neighbor, exactly conserving (to round-off error) the quantity $\mathcal{U}$.

Introduction to CFD: conservation versus accuracy

This cancellation of the sums is an example of a *telescoping property*, and is the main reason why finite-volume methods are attractive—conserved quantities are conserved to machine (roundoff) precision.

Note that conservation is not the same as accuracy. We can construct the fluxes for our discretized equation by calling a random number generator and we'd still be conservative, but not at all accurate.

Introduction to CFD: boundary conditions

3.3.3 Boundary conditions with finite-volume grids

Imagine that we wish to compute the derivative in every zone using:

$$\left.\frac{\partial f}{\partial x}\right|_i = \frac{f_i - f_{i-1}}{\Delta x} \quad . \tag{3.15}$$

If we denote the index corresponding to the leftmost zone in our domain as 'lo', then when we try to compute $\partial f/\partial x|_{\rm lo}$, we will "fall-off" the grid, i.e., we need a value of f for zone $\rm lo - 1$, which is outside the domain. This is where boundary conditions for our grid come into play.

Introduction to CFD: boundary conditions

We implement boundary conditions by extending the computational grid beyond the physical domain (see Figure 3.2). These additional zones are called *ghost cells*. They exist solely to handle the boundary conditions and allow us to use the same update equation (e.g. Eq. 3.15) for all zones in the domain.

Figure 3.2: A simple 1-d finite-volume grid with a single ghost cell at each end of the domain. The domain boundaries are indicated by the heavy vertical lines. Here there are hi − lo + 1 zones used in the discretization of the domain, with the first zone in the domain labeled lo and the last in the domain labeled hi.

Introduction to CFD: boundary conditions

The number of ghostcells needed for the grid depends on how wide the stencils used in the computation are. The wider the stencil, the more ghostcells are needed.

Periodic boundary conditions would be implemented as:

$$f_{\mathrm{hi}+1} = f_{\mathrm{lo}} \tag{3.16}$$

$$f_{\mathrm{lo}-1} = f_{\mathrm{hi}} \tag{3.17}$$

A simple outflow (zero-gradient) boundary condition would be implemented as:

$$f_{\mathrm{hi}+1} = f_{\mathrm{hi}} \tag{3.18}$$

$$f_{\mathrm{lo}-1} = f_{\mathrm{lo}} \tag{3.19}$$

3.4 Numerical implementation details

The computational grids used for finite-volume techniques maps nicely onto multi-dimensional arrays. For the cell-average data, a typical array would be dimensioned as

```
ilo = ng
ihi = nx + ng - 1
state = np.zeros((nx + 2*ng, nvar))
```

in python (using NumPy), where nx is the number of zones in the domain and ng is the number of ghost cells.

Introduction to CFD: boundary conditions

For some problems, there might be more than one state variable, so `nvar`, is the number of state variables that live in each zone. We would then loop from `ilo` to `ihi` to access the state data in the grid interior.

Fluxes are also stored in arrays, but since we cannot index an array with half-integer indices, the standard convention is that the index `i` in an array refers to the flux on the *left* edge of that zone. I.e.,

$$\texttt{flux[i]} \quad \longleftrightarrow \quad F_{i-1/2}$$

Introduction to CFD: Eulerian versus Lagrangian frames

It is useful to think about how our advected quantity, $a(x,t)$, changes in time. The full time derivative is:

$$\frac{da(x,t)}{dt} = \frac{\partial a}{\partial t} + \frac{\partial a}{\partial x}\frac{dx}{dt} \tag{4.19}$$

So the value of this derivative depends on the path, $x(t)$, that we choose to follow.

Consider an observer who is stationary. They will watch the flow move past them, so $dx/dt = 0$, and $da/dt = \partial a/\partial t$. This fixed frame is called *Eulerian frame*.

Introduction to CFD: Eulerian versus Lagrangian frames

Instead imagine an observer who moves with the flow, at the velocity u. This way they keep pace with an individual feature in the flow and track the changes it experiences. In this case, $dx/dt = u$, and our derivative, commonly written as D/Dt is:

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u\frac{\partial}{\partial x} \tag{4.20}$$

This is the *Lagrangian frame*, and the derivative, D/Dt is called the *Lagrangian derivative*, *material derivative*, *convective derivative*, or *advective derivative*[†].

Our linear advection equation can be written simply as $Da/Dt = 0$. We've been solving the equations in the Eulerian frame—our grid is fixed and the fluid moves through it. For hydrodynamics, it will be useful conceptually to consider the Lagrangian frame to understand how the fluid properties change in a particular fluid element over time.

Introduction to CFD: Errors and convergence rate

For the advection problem (with $u > 0$), the analytic solution is to simply propagate the initial profile to the right. This means that with periodic boundary conditions, after advecting for one period, our numerical solution should be identical to the initial conditions. Any differences are our numerical error. We can quantify the error by taking the norm of error‡ as:

$$\epsilon^{\text{abs}} = \|a^{\text{final}} - a^{\text{init}}\|_2 \equiv \left[\frac{1}{N}\sum_{i=1}^{N}(a_i^{\text{final}} - a_i^{\text{init}})^2\right]^{1/2} \tag{4.21}$$

It is sometimes useful to compare to the norm of the original solution to get a measure of the relative error:

$$\epsilon^{\text{rel}} \equiv \frac{\|a^{\text{final}} - a^{\text{init}}\|_2}{\|a^{\text{init}}\|_2} \tag{4.22}$$

Note that for the absolute norm, it is important in these definitions to normalize by the number of zones, N, otherwise our error will be resolution-dependent. For the relative norm, since we scale by a norm on the same grid, this normalization will cancel.