Homework 4 Notes

Due: Oct 12, 2018

1 Computing Errors

When your numerical scheme is solving for a function $f(x)$ instead of a single point like $x(t)$, it may not be obvious at first how to calculate errors or compare to an exact solution.

Fortunately, its a simple extension of the L_1 scheme we have already used in the class. Analytically, your code returns some representation of the solution at time t on the domain $[a, b]$, call it $f_{num}(x; t)$. There is also of course the exact solution $f_{exact}(x; t)$. The L_1 error of f_{num} is:

$$L_1(f_{num}) = \int_a^b |f_{num}(x;t) - f_{exact}(x;t)| \, \mathrm{d}x \ . \tag{1}$$

This is a special case of the p-norm, which is written:

$$L_p(f, g) = \int_a^b |f(x)^p - g(x)^p|^{1/p} \, \mathrm{d}x \ . \tag{2}$$

In a hydrodynamics code, you are solving for three functions ρ, v, and P. You can calculate the error for any one of them, or for a combination like the entropy $s = s(\rho, P)$.

How you compute the integral (1) with your numerical data depends on your scheme. For our hydrodynamics code, the grid is separated into N_x cells of width $\Delta x = (b - a)/N_x$. The boundaries between cells are at $x_{i-1/2} = a + i\Delta x$, where $i = 0, \ldots, N$. The cell centres are at $x_i = a + (i + 1/2)\Delta x$, where $i = 0, \ldots, N - 1$.The discrete fluid variables (e.g. ρ) are *averages* over each cell:

$$\rho_i \equiv \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} \rho(x) \, \mathrm{d}x \ . \tag{3}$$

To second order (the maximum order of this code), this is equivalent to evaluating the variable in the *center* of the cell:

$$\rho_i = \rho\left(x_i\right) + \mathcal{O}\left(\Delta x^2\right) \ . \tag{4}$$

Then to the same order, we can calculate the integral (1) using a centred Riemann sum:

$$L_1(\rho_{num}) = \Delta x \sum_{i=0}^{N-1} \left|\rho_i - \rho_{exact}\left(x_i\right)\right| \ . \tag{5}$$

For a grid on $[0, 1]$ with 10 zones, the cell centres are at $x_i = \{0.05, 0.15, \ldots, 0.95\}$ and the interfaces at $x_{i-1/2} = \{0, 0.1, \ldots, 0.9, 1.0\}$.

2 Boundary Conditions

The time evolution for any particular zone i depends on its neighbours $i - 1$ and $i + 1$ in a first order scheme. For a second order scheme it also depends on $i - 2$ and $i + 2$. This causes

trouble for zones 0, 1, $N-2$, and $N-1$, as they don't have enough neighbours! This is the most pragmatic reason to consider the need for boundary conditions.

Most boundary schemes are implemented using *ghost zones*: numerical cells outside the domain of the calculation. We can keep all the math of previous section, and consider ghost zones by letting the cell index $i = -1, 0, 1, \ldots, N-2, N-1, N$ for one ghost zone or $i = -2, -1, 0, \ldots, N-1, N, N+1$ for two ghost zones. In general, for N_g ghost zones, $i = -N_g, \ldots, N + N_g - 1$.

Ghost zones should not be updated in the same manner as the regular zones. They only exist to provide inputs for the evolution of regular zones. Two common ways to set boundaries are Dirichlet and Neumann.

2.1 Dirichlet

For a function $f(x,t)$ on $[a,b]$ Dirichlet boundaries specify $f(a,t)$ and $f(b,t)$ for all t. Often these zones are fixed to the initial condition $f(a,t) = f(a,0)$ and $f(b,t) = f(b,0)$. To do this in a code, simply set your ghost zones initially and never update them. For the variable ρ_i denote $\rho(x_i, t_n)$ as ρ_i^n. Then fixing to the initial condition would look like:

$$\rho^n_{-2} = \rho^0_{-2} \ , \qquad \rho^n_{-1} = \rho^0_{-1} \ , \qquad \rho^n_N = \rho^0_N \ , \qquad \rho^n_{N+1} = \rho^0_{N+1} \tag{6}$$

2.2 Neumann

For a function $f(x,t)$ on $[a,b]$ Neumann boundaries specify $\partial_x f(a,t)$ and $\partial_x f(b,t)$ for all t. A common choice is to fix $\partial_x f(a,t) = \partial_x f(b,t) = 0$. This resembles the free end of an oscillating string, and allows waves to freely move out of the domain. It is commonly called *outflow*. On our grid for the variable ρ, to first order we can write the derivative at $x = a$ as:

$$\partial_x \rho(a,t) \approx \frac{\rho(x_0,t) - \rho(x_{-1},t)}{\Delta x} = \frac{\rho^n_0 - \rho^n_{-1}}{\Delta x} \tag{7}$$

For the outflow condition $\partial_x \rho(a,t) = 0$ we can rearrange this to find ρ^n_{-1}:

$$\rho^n_{-1} = \rho^n_0 \ . \tag{8}$$

Similarly at $x = b$:

$$\rho^n_N = \rho^n_{N-1} \ . \tag{9}$$

This is also called the *copy* boundary condition. For second order outflow, it is sufficient to continue the copying to the other ghost zones.

$$\rho^n_{-2} = \rho^n_{-1} = \rho^n_0 \qquad \rho^n_{N+1} = \rho^n_N = \rho^n_{N-1} \tag{10}$$

Notice for outflow boundaries the ghost zones are time dependent! They change with whatever values cells $i = 0, N-1$ take, and must be re-calculated every time the interior zones are updated.