1-D swept equations

Daniel Magee

August 2016

1 Heat

The heat equation without volumetric heat flux:

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T \tag{1}$$

In 1-D:

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2} \tag{2}$$

Discretizing with forward differencing in time and central differencing in space:

$$\frac{T_i^{m+1} - T_i^m}{\Delta t} = \alpha \frac{T_{i+1}^m + T_{i-1}^m + 2T_i^m}{\Delta x^2} \tag{3}$$

Defining the Fourier number $Fo = \frac{\alpha \Delta t}{\Delta x^2}$, and solving for temperature at the next timestep T_i^{m+1}:

$$T_i^{m+1} = Fo(T_{i+1}^m + T_{i-1}^m) + (1 - 2Fo)T_i^m \tag{4}$$

This is a first-order, explicit, finite-difference method.

With insulated boundary conditions at both ends and n spatial points:

$$T_0 = T_1 \quad and \quad T_{n+1} = T_{n-1} \tag{5}$$

2 Kuramoto-Sivashinsky

The Kuramoto-Sivashinsky equation is a non-linear, fourth-order, one-dimensional partial differential equation that, "generally describes the dynamics near long-wave-length primary instabilities in the presence of appropriate (translational, parity, and Galilean) symmetries."

$$u_t = -(uu_x + u_{xx} + u_{xxxx}) = -(\frac{1}{2}u_x^2 + u_{xx} + u_{xxxx}) \tag{6}$$

Discretizing the fourth derivative requires neighbors of neighbors in the spatial dimension or an additional step where u_{xx} is calculated for the three points in the domain.

Finite difference discretization with neighbors of neighbors:

$$\frac{u_i^{m+1} - u_i^m}{\Delta t} = -\left(\frac{(u_{i+1}^m)^2 - (u_{i-1}^m)^2}{4\Delta x} + \frac{u_{i+1}^m + u_{i-1}^m - 2u_i^m}{\Delta x^2} + \frac{u_{i+2}^m - 4u_{i+1}^m + 6u_i^m - 4u_{i-1}^m + u_{i-2}^m}{\Delta x^4}\right) \tag{7}$$

Finite difference discretization with flux step:

$$u_{xx} + u_{xxxx} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^4 u}{\partial x^4} = \frac{\partial^2}{\partial x^2} u + \frac{\partial^2}{\partial x^2} u_{xx} = \frac{\partial^2}{\partial x^2}\left(u + u_{xx}\right) \tag{8}$$

$$(u_{xx})_i = \frac{u_{i-1} + u_{i+1} - 2u_i}{\Delta x^2} \;\; at \;\; i-1,\; i,\; i+1 \tag{9}$$

$$\frac{u_i^{m+1} - u_i^m}{\Delta t} = -\left(\frac{(u_{i+1}^m)^2 - (u_{i-1}^m)^2}{4\Delta x} + \frac{(u + u_{xx})_{i+1}^m + (u + u_{xx})_{i-1}^m - 2(u + u_{xx})_i^m}{\Delta x^2} \right) \tag{10}$$

Let the right hand side be $f(u(x,t))$. Using a second-order, explicit Runge-Kutta method the solution at the new timestep is calculated in two steps. First the predictor solution is obtained at $u_i^{m+1/2}$.

$$u_i^{m+1/2} = u_i^m + \frac{\Delta t}{2} f(u_i^m) \tag{11}$$

Then it is evaluated and added to u_i^m to obtain u_i^{m+1}.

$$u_i^{m+1} = u_i^m + \Delta t f(u_i^{m+1/2}) \tag{12}$$

This requires the predictor step to be evaluated at all spatial points before the full timestep can be completed.
The boundary condition is periodic so, for example, for n spatial points $i+1$ is actually $mod((i+1), n)$.

3 Euler (Sod Shock Tube)

The Sod Shock Tube is a 1-D compressible flow problem based on the conservative form of the Euler equation shown below.

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} = 0 \tag{13}$$

$$Q = \begin{Bmatrix} \rho \\ \rho u \\ \rho e \end{Bmatrix} \qquad F = \begin{Bmatrix} \rho u \\ \rho u^2 + P \\ u(\rho e + P) \end{Bmatrix} \tag{14}$$

$$P = (\gamma - 1)(e - \frac{\rho u^2}{2}) \tag{15}$$

The boundary conditions are constant values for the state variables on either side of a diaphragm separating two parcels of the same fluid.

$$\begin{Bmatrix} \rho_L \\ u_L \\ P_L \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \\ 0.0 \end{Bmatrix} \qquad \begin{Bmatrix} \rho_R \\ u_R \\ P_R \end{Bmatrix} = \begin{Bmatrix} 0.125 \\ 0.1 \\ 0.0 \end{Bmatrix} \tag{16}$$

The equation is discretized using a second-order finite-volume scheme with cell centered values. The values are assumed to be constant over the length of the cell. The first step in the solution is calculating the pressure ratio of the current discrete volume and it's neighbor volumes. This is used to assess the need for and compute reconstructed fluxes at cell interfaces.

$$P_{r,i} = \frac{P_{i+1} - P_i}{P_i - P_{i-1}} \;\; at \;\; i-1,\; i,\; i+1 \tag{17}$$

This requires a five point stencil centered on Q_i. Left and right sided values are then obtained at the interfaces: $Q_{i-1/2}, Q_{i+1/2}$:

$$Q_{i-1/2}^L = \begin{cases} Q_{i-1} + \frac{\min(P_{r,i-1},1)}{2} * (Q_i - Q_{i-1}), & \text{if } 0 < P_{r,i-1} < \infty \\ Q_{i-1}, & \text{else} \end{cases} \tag{18}$$

$$Q_{i-1/2}^R = \begin{cases} Q_i + \frac{\min(P_{r,i}^{-1},1)}{2} * (Q_{i-1} - Q_i), & \text{if } 0 < \frac{1}{P_{r,i}} < \infty \\ Q_i, & \text{else} \end{cases} \tag{19}$$

$$Q^L_{i+1/2} = \begin{cases} Q_i + \frac{\min(P_{r,i},1)}{2} * (Q_{i+1} - Q_i), & \text{if } 0 < P_{r,i} < \infty \\ Q_i, & \text{else} \end{cases} \tag{20}$$

$$Q^R_{i+1/2} = \begin{cases} Q_{i+1} + \frac{\min((P_{r,i+1})^{-1},1)}{2} * (Q_i - Q_{i+1}), & \text{if } 0 < \frac{1}{P_{r,i+1}} < \infty \\ Q_{i+1}, & \text{else} \end{cases} \tag{21}$$

Once we have the reconstructed values on either side of each interface, we can calculate the flux at the respective boundaries. To show the formula more concisely, first calculate the spectral radius. Start by finding the spectral values at each interface. In these cases u and e can be found by dividing the left and right conservative values in Q by the first value of Q which is ρ.

$$Q_{sp} = \left\{ \begin{array}{c} \sqrt{\rho_L * \rho_R} \\ \frac{\sqrt{\rho_L} * u_L + \sqrt{\rho_R} * u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\ \frac{\sqrt{\rho_L} * e_L + \sqrt{\rho_R} * e_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \end{array} \right\} \tag{22}$$

Then find the spectral pressure P_{sp} with the values of spectral Q and the spectral radius is then:

$$r_{sp} = \sqrt{\frac{\gamma * P_{sp}}{Q_{sp,1}}} + |Q_{sp,2}| \tag{23}$$

The numbers in subscripts denote the index of the variable in Q_{sp} with one based indexing. The spectral radius is a scalar factor applied to the overall flux from the difference between values on either side of the interface. The other part of the overall flux is found by evaluating the function F across the interface. The overall flux is computed:

$$Flux = \frac{r_{sp}}{2} * (Q^L - Q^R) * (F^R + F^L) \tag{24}$$

The function $F(Q)$ is given in equation 14. This procedure is carried out at $x = i - 1/2$ and $x = i + 1/2$. Finally, a step forward is taken and the scheme is complete.

$$\text{Predictor Step: } Q_i^{n+1/2} = \frac{\Delta t}{2\Delta x}(Flux_{i+1/2} - Flux_{i-1/2}) \tag{25}$$

$$\text{Final Step: } Q_i^{n+1} = Q_i^{n+1/2} + \frac{\Delta t}{\Delta x}(Flux_{i+1/2} - Flux_{i-1/2}) \tag{26}$$

Effective computational optimizations for this scheme depend on the architecture and language. Generally a program can be accelerated by: storing only the most recent predictor and final values of a step in the most immediately accessible memory bank and pushing or flushing the past values, avoiding division by evaluating the properties of the numerator and denominator for the limitor conditions rather than computing P_r when it may not be used, storing $\Delta t/\Delta x$ as a global or constant variable instead of recalculating, and arranging the order computation to accentuate instruction level parallelism.