The (1D) Euler Density-Velocity Equation (E3)

This module allows you to do a numerical study of the leap frog and Lax-Wendroff numerical techniques applied to (1D) Euler density-velocity equation (E3)

$$\frac{\partial}{\partial t}(\rho u) + \frac{\partial}{\partial x}(u^2\rho + c^2\rho) = 0; -L < x < L$$

$$\frac{\partial}{\partial t}\rho + \frac{\partial}{\partial x}(\rho u) = 0; -L < x < L$$

with initial condition

$$\rho(x,0) = q_\rho \arctan(s_\rho * x) + r_\rho$$

$$u(x,0) = q_u \arctan(s_u * x) + r_u.$$

In the codes comprising this module you are asked to input L, rho_left (the value for $\rho(x,0)$ at $-\infty$) and rho_right (the value for $\rho(x,0)$ at ∞) and u_left (the value for $u(x,0)$ at $-\infty$) and u_right (the value for $u(x,0)$ at ∞). You also input the time step, dt for t and the grid size, dx for x.

There are Maple routines, Matlab routines and Fortran 90 codes for running the numerical algorithms. The Maple routines are

- density-velocity-lf.mws (conservation form using the leap frog finite difference scheme)
- density-velocity-lw.mws (conservation form using the Lax Wendroff finite difference scheme)

The Matlab routines are

- densityvelocitylf.m (conservation form using the leap frog finite difference scheme)
- densityvelocitylw.m (conservation form using the Lax Wendroff finite difference scheme)

The Fortran 90 codes are

- density-velocity-lf.f90 (conservation form using the leap frog finite difference scheme)
- density-velocity-lw.f90 (conservation form using the Lax Wendroff finite difference scheme)

Choose one or more of the above routines and run the two codes using u_left = -1, u_right = 1, s_u = 1,rho_left = 1, rho_right = 1, s_rho = 1, L = 10, T = 50*dt, dt = 0.03125 and dx = 0.25.

Convince yourself using graphics that the results you are getting are reasonable. The Maple and Matlab routines will do the graphics for you. The Fortran 90 codes output data sets. These data sets have the form name00010. The 00010 represents the 10th time step. For the Fortran 90 codes you have to load the data sets into a graphics visualizer.

Now let u_left = 1, u_right = -1, s_u = 1,rho_left = 1, rho_right = 1, s_rho = 1, L = 10, T = 50*dt, dt = 0.03125 and dx = 0.25. Run the codes and compare with the previous results.

Now try u_left = 0, u_right = 0, s_u = 1,rho_left = 1, rho_right = 2, s_rho = 1, L = 10, T = 100*dt, dt = 0.03125 and dx = 0.25. Run the codes and compare with the previous results.

Now make L larger in the first input set and see what happens. Make s_u larger and see what happens. Make dt larger. What happens? Now make dt smaller and see what happens. You should ask how these tests relates to the stability of the numerical algorithm.

Now experiment by changing u_left , u_right, s_u ,rho_left , rho_right , s_rho , dt and dx. What results do you find? Can you explain these physically and numerically?

From the equation of state plot the pressure profiles in one of the codes. Take the leap frog and Lax-Wendroff codes in one of the software packages and include diffusion (νu_{xx} in the momentum equation). Run the above tests and see what happens.