Parallel Process Final Project Proposal
Name: Max Le
Project Title: Parallelization Of The 1D Shock Tube

1 Problem Description

Shockwaves occurs in nature when we have an object that moves faster than the speed of sound. Usually, this happens due to a discontinuity in the air when the object goes from subsonic to supersonic. This class of problem belongs to the hyperbolic PDEs and are fascinating because they can be applied to many applications such as supersonic combustion and aircraft designs. I am going to simulate this phenomena in a much simpler case: the shock tube. This is a simple mechanical device with gases of different initial densities and pressures on both side, and a diaphragm separating the two regions. Once the diaphragm is rupture, gas from the high pressure region will produce waves that enter the low pressure region. We will see a shockwave going to the lower pressure region, and an expansion wave going back to the higher pressure region. Numerically, this simple problem is governed by the three conservation laws: mass, momentum, and energy. In terms of the flux vector F and state vector U, this is the equation that I am going to solve:

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} = 0 \tag{1}$$

2 Serial Algorithm

I am going to use a 1st order upwind scheme with either Steger-Warming flux or Van Leer flux to solve. In particular, we are going to employ a flux splitting scheme. The time derivative will be approximated as a forward difference; while the spatial derivatives are approximated as forward difference (if speed of sound is negative) and backward difference (if speed of sound is positive). Initial conditions will be given for both sides of the tube. We are going to assume this is 1D, and also ideal gas. In discretized form, the equation looks like this, n is time level and i is nodal point.

$$U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x}\left[(F^+ + F^-)_{i+1/2} - (F^+ + F^-)_{i-1/2}\right] \tag{2}$$

Of course, the nature of the speed of sound will dictate when we use F^+ (upwind) and F^- (downwind)

3 Parallelization Strategy

Because this is a time marching problem in its simplest form, I am going to attempt to use MPI to send and receive data from the previous time, so that these data can be used in the updated solution. We are going to try to minimize the number of communications. In other words, instead of having a series of MPI send/receive at every grid points, we can try to have at every 4,8,16,32 grid points. The challenge here would be that instead of a single array that keeps getting updated; we have to worry about a 3xN matrix (the U state vector) and also how to send/receive in the case of upwind/downwind.

4 Investigative Questions

The interesting trend to investigate would be that how far can we go in terms of minimizing the communications. Additionally, another obvious question to investigate would be in the speed up and problem size.