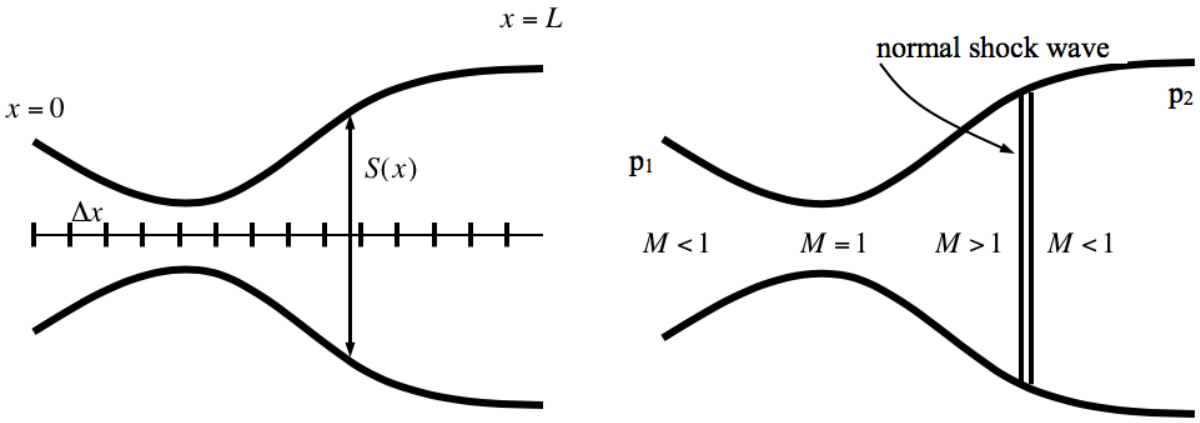
AEM 4253/5253 – Computational Fluid Mechanics (Fall 2018)

TERM PROJECT
Due: Wednesday, December 12th
(no late projects will be accepted)

A Quasi-One Dimensional Flow Solver Applied to Nozzles and Shock-Tubes

The goal of this project is to write a computer program capable of simulating the steady-state operation of a general subsonic, transonic, supersonic nozzle as well as the unsteady operation of a shock-tube. The physical model used will be the Navier-Stokes equations, including the energy equation, but neglecting viscous terms (the Euler Equations) assuming a caloric and thermally perfect gas.

The Quasi-1D Nozzle Problem:



A "Laval" nozzle operates under a pressure differential. If $p_2 < p_1$ gas will flow through the nozzle, it will accelerate as the cross-sectional area (S) decreases and decelerate as the area increases. If p_2 is sufficiently low enough, the gas will reach sonic speed (M=1) at the throat and continue to accelerate (M>1) in the diverging portion of the nozzle. Depending on the precise value of p_2 , a normal shock may sit in the nozzle; termed a transonic nozzle. The problem is quasi-one-dimensional in that the properties at each x-location can be assumed constant over the entire cross-section. However, dealing with the range of Mach numbers and accurately 'capturing' a shock wave requires a general and somewhat sophisticated CFD method.

Nozzle Geometry:

$$S(x) = \begin{cases} 1 + \frac{3}{2}\left(1 - \frac{1}{5}x\right)^2, & 0 \le x \le 5 \\ 1 + \frac{1}{2}\left(1 - \frac{1}{5}x\right)^2, & 5 < x \le 10 \end{cases}$$

NOTE: You will need to evaluate dS/dx. Do so analytically, just take the derivative!

CASE 1 (Subsonic Nozzle)

$\rho_1 = 1.1409 \ \text{kg/m}^3$ $\quad \rho_2 = 1.1008 \ \text{kg/m}^3$

$u_1 = 65.451 \ \text{m/s}$ $\quad u_2 = 113.060 \ \text{m/s}$ $\quad$ or

$p_1 = 9.7534 \times 10^4 \ \text{Pa}$ $\quad p_2 = 9.2772 \times 10^4 \ \text{Pa}$

$$\begin{cases} p_{total} = 1.0 \times 10^5 \ \text{Pa} \\ T_{total} = 300 \ \text{K} \\ p_{back} = 9.2773 \times 10^4 \ \text{Pa} \end{cases}$$

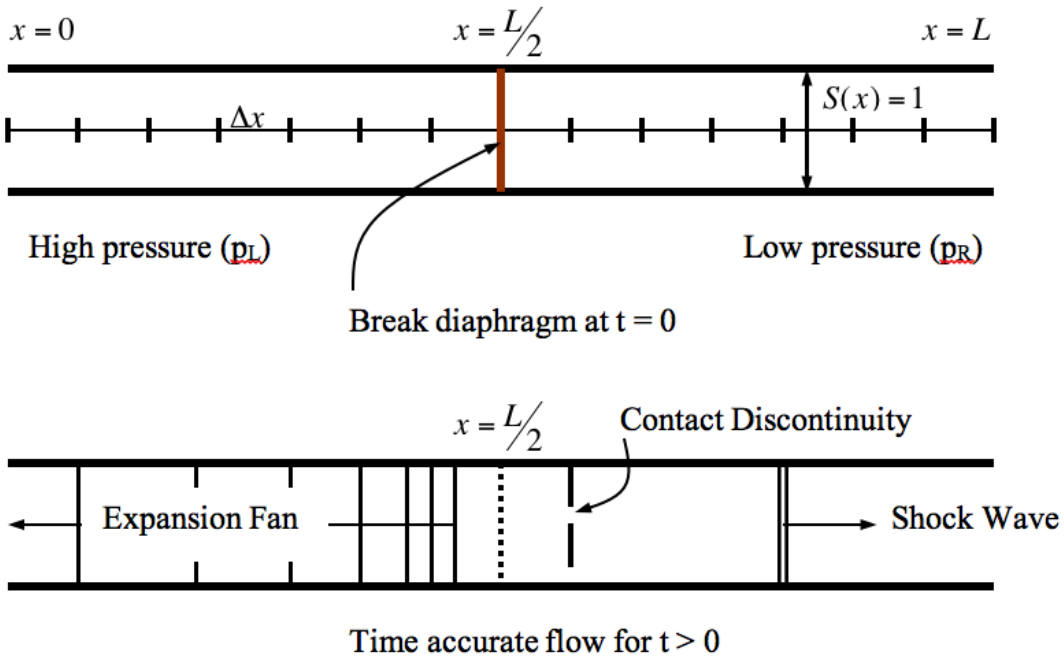
CASE 2 (Transonic Nozzle)

$\rho_1 = 1.1308 \ \text{kg/m}^3$ $\quad \rho_2 = 1.0275 \ \text{kg/m}^3$

$u_1 = 80.054 \ \text{m/s}$ $\quad u_2 = 150.535 \ \text{m/s}$ $\quad$ or $\quad$ $\begin{cases} p_{total} = 1.0 \times 10^5 \ \text{Pa} \\ T_{total} = 300 \ \text{K} \\ p_{back} = 8.4974 \times 10^4 \ \text{Pa} \end{cases}$

$p_1 = 9.6328 \times 10^4 \ \text{Pa}$ $\quad p_2 = 8.4974 \times 10^4 \ \text{Pa}$

The Shock-Tube Problem:



By using the same physical model and simply setting S(x)=1, one can use the same program to simulate the operation of shock-tube. In a real shock-tube, a diaphragm separates a high-pressure gas from a low-pressure gas. When the diaphragm is removed (punctured) at t = 0, a shock wave propagates in one direction, an expansion fan propagates in the other, and a contact discontinuity may also be present inside the tube.

CASE 3 (Shock-Tube)

$$\left.\begin{array}{c}\rho_L = 1.0 \text{ kg/m}^3 \\ u_L = 0.0 \text{ m/s} \\ p_L = 1.0\times10^5 \text{ Pa}\end{array}\right\} 0 \le x \le 5 \qquad \left.\begin{array}{c}\rho_R = 0.125 \text{ kg/m}^3 \\ u_R = 0.0 \text{ m/s} \\ p_R = 1.0\times10^4 \text{ Pa}\end{array}\right\} 5 < x \le 10 \qquad S(x) = 1$$

Non-dimensional Variables **(Optional)**:

It is often good-practice to work with non-dimensional 'normalized' variables. Doing so can help avoid operating on very small numbers (can cause roundoff errors in some programming languages). If so, for all numerics, redefine:

$$\rho = \frac{\rho}{\rho_1} \quad , \quad u = \frac{u}{a_1} \, , \quad \rho E = \frac{\rho E}{\gamma p_1} , \quad \text{where } a_1 = \sqrt{\gamma R T}_1$$

When presenting results, be sure to convert back to dimensional variables.

The Governing Equations

For inviscid, compressible flow of a thermally and calorically perfect gas (air), the governing equations applied to a quasi-1D problem have the following form:

$$\frac{\partial Q}{\partial t} + \frac{\partial F(Q)}{\partial x} = g$$

$$Q = \begin{bmatrix} \rho S \\ \rho u S \\ \rho E S \end{bmatrix} \qquad F = \begin{bmatrix} \rho u S \\ (\rho u^2 + p) S \\ \rho u H S \end{bmatrix} \qquad g = \begin{bmatrix} 0 \\ p \frac{dS}{dx} \\ 0 \end{bmatrix}$$

$$p = (\gamma - 1)(\rho E - \tfrac{1}{2}\rho u^2) \qquad H = E + \tfrac{p}{\rho} = c_V T + \tfrac{1}{2} u^2 + \tfrac{p}{\rho}$$

Numerical Method (Yee-Roe Flux Limited Scheme)

The numerical method to be used is an upwind scheme. The scheme is made 2nd – order accurate in space via a specific limiter proposed by Yee [1], which reverts back to 1st - order near discontinuities in order to maintain stability. An interesting aspect of the method is that Yee's limiter is actually applied to the characteristic variables themselves. To further deal with large discontinuities between nodes, Roe's approximate Reimman solution [2] is used to determine the flux variables at cell interfaces. The Explicit Euler method will be used to iterate the solution in time.

When using Explicit Euler time marching, the numerical method is written compactly as:

$$Q_i^{n+1} = Q_i^n - \Delta t\left(\left[\delta_x F_i\right] - g\right)$$

$$\left[\delta_x F_i\right] = \frac{1}{2\Delta x}\left[F_{i+1} - F_{i-1} - (R^{-1}|\lambda|N)_{i+\frac{1}{2}} + (R^{-1}|\lambda|N)_{i-\frac{1}{2}}\right]$$

$$N_i = M_i - \text{ minmod}(M_{i-1}, M_i, M_{i+1})$$

$$M_i = R_i(Q_{i+\frac{1}{2}} - Q_{i-\frac{1}{2}})$$

$|\lambda_i| \equiv$ absolute value of the eigenvalues of the Flux Jacobian $\dfrac{\partial F_i}{\partial Q_i}$

$R_i^{-1} \equiv$ left eigenvectors of the Flux Jacobian $\dfrac{\partial F_i}{\partial Q_i}$

$R_i \equiv$ right eigenvectors of the Flux Jacobian $\dfrac{\partial F_i}{\partial Q_i}$

$[\ \ ]_{i\pm\frac{1}{2}}$ means evaluation at the Roe Average State

The **minmod()** function returns the minimum of its arguments if the arguments are all positive, the maximum if its arguments are all negative, and zero if its arguments are of mixed sign. Since ***M*** is a 3x1 matrix, the **minmod()** function should be performed on each row of ***M*** independently.

NOTE: If you inspect this flux function, you should see that it cannot be evaluated at node (2) or node (N-1) as it requires information at (i-2) and (i+2). As a temporary solution, you can ignore the minmod() function for nodes (2) and (N-1). Once your code is working, you can go back and deal with this problem more appropriately.

Roe's Approximate Reimann Solution

Often, as for the current method, the vector of flux variables must be evaluated at the interface between two cells. In order to compute the flux variables, an 'average' state must be determined at the interface. Roe derived the following approximate-Reimman solution for the average state:

$$\overline{\rho} = \sqrt{\rho^R \rho^L}$$

$$\overline{u} = \frac{\sqrt{\rho^L}u^L + \sqrt{\rho^R}u^R}{\sqrt{\rho^L} + \sqrt{\rho^R}}$$

$$\overline{H} = \frac{\sqrt{\rho^L}H^L + \sqrt{\rho^R}H^R}{\sqrt{\rho^L} + \sqrt{\rho^R}}$$

Thus: for $\overline{\rho} = \rho_{i+\frac{1}{2}}$ then $\begin{cases} \rho^L = \rho_i \\ \rho^R = \rho_{i+1} \end{cases}$, and for $\overline{\rho} = \rho_{i-\frac{1}{2}}$ then $\begin{cases} \rho^L = \rho_{i-1} \\ \rho^R = \rho_i \end{cases}$.

The same applies for the other variables.

Note that the Flux Jacobian and it's components can all be expressed in function of the above 3 variables. That is:

$$\overline{A} = fcn(\overline{u}, \overline{H}) \qquad \overline{R} = fcn(\overline{\rho}, \overline{u}, \overline{a}) \quad \overline{R}^{-1} = fcn(\overline{\rho}, \overline{u}, \overline{a}) \qquad \overline{\lambda} = fcn(\overline{u}, \overline{a})$$

where $\overline{a} = fcn(\overline{u}, \overline{H})$

The variables above are defined as follows:

$$\left|A\right| \equiv R^{-1}\left|\Lambda_A\right|R$$

$$A = \frac{\partial F}{\partial Q} = \begin{bmatrix} 0 & 1 & 0 \\ \frac{\gamma-3}{2}u^2 & -(\gamma-3)u & (\gamma-1) \\ -Hu + \frac{(\gamma-1)}{2}u^3 & H-(\gamma-1)u^2 & \gamma u \end{bmatrix}$$

$$R^{-1} = \begin{bmatrix} 1 & \frac{\rho}{\sqrt{2}a} & \frac{\rho}{\sqrt{2}a} \\ u & \frac{\rho}{\sqrt{2}a}(u+a) & \frac{\rho}{\sqrt{2}a}(u-a) \\ \frac{u^2}{2} & \frac{\rho}{\sqrt{2}a}\left[\frac{u^2}{2} + \frac{a^2}{(\gamma-1)} + au\right] & \frac{\rho}{\sqrt{2}a}\left[\frac{u^2}{2} + \frac{a^2}{(\gamma-1)} - au\right] \end{bmatrix}$$

$$R = \begin{bmatrix} 1 - \frac{(\gamma-1)}{2}\frac{u^2}{a^2} & (\gamma-1)\frac{u}{a^2} & \frac{-(\gamma-1)}{a^2} \\ \frac{1}{\sqrt{2}\rho a}\left[\frac{(\gamma-1)}{2}u^2 - au\right] & \frac{1}{\sqrt{2}\rho a}\left[a-(\gamma-1)u\right] & \frac{1}{\sqrt{2}\rho a}(\gamma-1) \\ \frac{1}{\sqrt{2}\rho a}\left[\frac{(\gamma-1)}{2}u^2 + au\right] & -\frac{1}{\sqrt{2}\rho a}\left[a+(\gamma-1)u\right] & \frac{1}{\sqrt{2}\rho a}(\gamma-1) \end{bmatrix}$$

$$\Lambda_A = \begin{bmatrix} u & 0 & 0 \\ 0 & u+a & 0 \\ 0 & 0 & u-a \end{bmatrix}$$

$$a^2 = (\gamma-1)\left[H - \tfrac{1}{2}u^2\right]$$

Project Report Format (required sections and specific questions)

1.0 Introduction

- provide a brief overview of the problems and relevant governing equations (~2 pages)

2.0 Numerical Method

- list the numerical method in compact form (listing Roe average equations and characteristic matrices is not necessary)
- provide an explanation as to why such a method is suitable for the problems considered (be sure to refer to concepts discussed in class) (~2 pages)

3.0 Numerical Results

3.1 Nozzle Simulations

- Explicit Euler (simulate **CASE 1** and **CASE 2**)
    - run to steady-state using a CFL=0.5, start with N=10 and increase N until sufficient grid-convergence is achieved (thus decide on an appropriate N)
    - plot the solution for a few values of N (~3 lines on one graph)
    - for chosen value of N, plot the solution at 2 times prior to steady-state, as well as the solution at steady-state (plot the 3 lines on one graph)
    - plot the 'convergence', that is, plot log(Residual) vs. timesteps
        - Residual = $\max_{over\text{-}x}[abs(Q^{n+1} - Q^n) / Q^0]$ – use $Q = \rho u$, (i.e. the percentage change in solution during each update)
    - in addition, for **CASE 2** only, run 2nd order without using the limiter
        - to do this, minmod() should return M_{i-1} only (2nd order upwind)
    - experiment with higher CFL values (comment on the stability)

3.2 Shock Tube Simulations

- Explicit Euler (simulate **CASE 3**)
    - run to a time where all flow features are distinct
    - determine a stable timestep (it will depend on N)
    - run for N=50,100, and 500 with 2nd order and limiter (plot results, discuss)
    - run for N=100 with 2nd order but no limiter (plot results, discuss)
    - run with N=100 with 1st order and no limiter (plot results, discuss)

4.0 Independent Study (worth 10% of final project grade)

- AFTER everything else is finished and working well, choose **one** option for further study. Various options include:
    (a) close both ends of the shock tube and watch the wave reflections!
    (b) implement real boundary conditions for the nozzle (since inflow and outflow is subsonic, what should really be specified is p_{total}, T_{total}, and p_{back}) – see me if interested in how to do this
    (c) it might be possible to connect a shock-tube to a nozzle and create your own virtual supersonic wind tunnel – see me if interested in how to do this
    (d) try a different method or a different limiter and compare certain results
    (e) something else you think of (keep it simple… it's only worth 10%)

5.0 Conclusions

- briefly summarize the major findings of the project (<1 page)

****Hand a print-out of your code.****