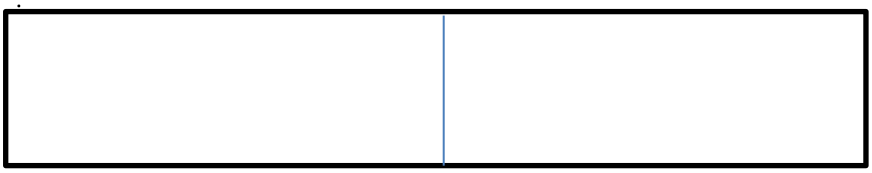
AUSM Flux Vector Splitting Scheme for Sod Shock Tube Problem

1. Problem



Initial condition:

Density_left = 1.0 Density_right = 0.125
Pressure_left = 1.0 Pressure_right = 0.1
Velocity_left = 0.0 Velocity_right = 0.0

2. Numerical scheme

1D Euler Equation in conservation law form is $\mathbf{U}_t + \mathbf{F}(\mathbf{U})_x = \mathbf{0}$

Where $\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ E \end{bmatrix}$, $\mathbf{F}(\mathbf{U}) = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ u(E+p) \end{bmatrix}$.

Splitting the flux vector **F** into convective component and pressure component,

$$\mathbf{F}(\mathbf{U}) = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u w \\ u(E+p) \end{bmatrix} = \begin{bmatrix} \rho u \\ \rho u^2 \\ \rho u v \\ \rho u w \\ \rho u H \end{bmatrix} + \begin{bmatrix} 0 \\ p \\ 0 \\ 0 \\ 0 \end{bmatrix} \equiv \mathbf{F}^{(c)} + \mathbf{F}^{(p)}$$

Where the convective flux component is $\mathbf{F}^{(c)}_{i+\frac{1}{2}} = M_{i+\frac{1}{2}} \left[\hat{\mathbf{F}}^{(c)} \right]_{i+\frac{1}{2}}$

The cell-interface Mach number is given by the splitting, $M_{i+\frac{1}{2}} = M_i^+ + M_{i+1}^-$

The splitting of the pressure flux component $p_{i+\frac{1}{2}} = p_i^+ + p_{i+1}^-$

Splitting of the Mach number,
$$M^{\pm}=\begin{cases}\pm\frac{1}{4}(M\pm1)^2 & \text{if } \mid M\mid\leq 1\\ \frac{1}{2}(M\pm\mid M\mid) & \text{if } \mid M\mid>1\end{cases}$$

Splitting of the pressure,
$$p^{\pm}=\begin{cases}\frac{1}{2}p(1\pm M) & \text{if } \mid M\mid\leq 1\\ \frac{1}{2}p\frac{(M\pm\mid M\mid)}{M} & \text{if } \mid M\mid>1\end{cases}$$

(An alternative choice is
$$p^{\pm}=\begin{cases}\frac{1}{4}p(M\pm1)^2(2\mp M) & \text{if } \mid M\mid\leq 1\\ \frac{1}{2}p\frac{(M\pm\mid M\mid)}{M} & \text{if } \mid M\mid>1\end{cases}$$
)

3. Source codes:

```fortran

!!!     This program solves Riemann problem for the Euler equations using AUSM first-order upwind methods
!!!     This work is licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License.
!!!     Ao Xu, Profiles: <http://www.linkedin.com/pub/ao-xu/30/a72/a29>

!!!
!!!                                 Shock Tube
!!!     -------------------------|-------------------------
!!!     |                             |                             |
!!!     |                             |                             |
!!!     |                             |                             |
!!!     -------------------------|-------------------------
!!!                                 Contact Discontinuity
!!!

!!!     x1,x1----------------Left/Right side
!!!     diaph----------------Initial discontinuity
!!!     pl,pr----------------Left/Right side pressure
!!!     rhol,rhor------------Left/Right side density
!!!     ul,ur----------------Left/Right side velocity
!!!     al,ar----------------Left/Right side local speed of sound

        program main
        implicit none
        integer, parameter :: N=1000
        integer :: i, itert, itc
        real(8), parameter :: gamma=1.4d0, R=287.14d0
        real(8) :: X(0:N+2), p(0:N+2), a(0:N+2), u(0:N+2), u1(0:N+2), u2(0:N+2), u3(0:N+2), f1(0:N+1), f2(0:N+1), f3(0:N+1)
```

```fortran
    real(8) :: x1, x2, dx, t, dt, lambda, diaph
    real(8) :: pl, pr, ul, ur, al, ar, rul, rur, retl, retr,rhol, rhor

    !!! input initial data
    x1 = 0.0d0
    x2 = 1.0d0
    dx = (x2-x1)/float(N)
    t = 0.25d0
    dt = 1e-4
    itert = NINT(t/dt)
    lambda = dt/dx
    diaph = 0.5d0
    itc = 0
    pl = 1.0d0
    pr = 0.1d0
    rhol = 1.0d0
    rhor =0.125d0
    ul = 0.0d0
    ur = 0.0d0
    al = SQRT(gamma*pl/rhol)
    ar = SQRT(gamma*pr/rhor)

    !!! convert primitive variables to conservative variables
!!!    rul,rur--------------Left/Right side   rho*u
!!!    retl,retr------------Left/Right side   rho*E = rho*(e+0.5*u^2)= rho*(p/rho/(gamma-1)+0.5*u^2) = p/(gamma-1)+0.5*u^2*rho
!!!    E = e+0.5*u^2            e = p/rho/(gamma-1)
    rul = rhol*ul
    rur = rhor*ur
    retl = 0.5d0*rul*ul+pl/(gamma-1.0d0)
    retr = 0.5d0*rur*ur+pr/(gamma-1.0d0)

    !!! construct initial conditions
    call initial(N,X,dx,diaph,rhol,rhor,rul,rur,retl,retr,u1,u2,u3)
    t = 0

    do itc=1,itert
        !!! find conservative numerical fluxes
        t = t + dt
        do i=0,N+1
            call AUSM(gamma,u1(i),u2(i),u3(i),u1(i+1),u2(i+1),u3(i+1),f1(i),f2(i),f3(i))
        enddo

```

```fortran
                !!! update conserved variables
                do i=1,N+1
                        u1(i) = u1(i)-lambda*(f1(i)-f1(i-1))
                        u2(i) = u2(i)-lambda*(f2(i)-f2(i-1))
                        u3(i) = u3(i)-lambda*(f3(i)-f3(i-1))
                        u(i) = u2(i)/u1(i)
                        p(i) = (gamma-1.0)*(u3(i)-0.5*u2(i)*u2(i)/u1(i))
                        a(i) = SQRT(gamma*p(i)/u1(i))
                enddo
        enddo

        write(*,*) 'AUSM first-order upwind methods'
        write(*,*) 'dt=',dt
        write(*,*) 'dx=',dx
        write(*,*) 'Final time = ',t

        open(unit=02,file='./shock_tube_AUSM.dat',status='unknown')
        write(02,101)
        write(02,102)
        write(02,103) N
        do i = 1,N+1
                        p(i) = (gamma-1.0d0)*(u3(i)-0.5d0*u2(i)*u2(i)/u1(i))
                        a(i) = SQRT(gamma*p(i)/u1(i))
                        u(i) = u2(i)/u1(i)
                        write(02,100) X(i), u1(i), p(i) ,u(i)
        enddo

100     format(2x,10(e12.6,'      '))
101     format('Title="Sod Shock Tube"')
102     format('Variables=x,rho,p,u')
103     format('zone',1x,'i=',1x,i5,2x,'f=point')

        close(02)
        write(*,*) 'Data export to ./shock_tube_AUSM.dat file!'

        stop
        end program main



        subroutine initial(N,X,dx,diaph,rhol,rhor,rul,rur,retl,retr,u1,u2,u3)
        implicit none
        integer :: i, N
```

```
        real(8) :: dx, diaph, rhol, rhor, rul, rur, retl, retr
        real(8) :: X(0:N+2),u1(0:N+2), u2(0:N+2), u3(0:N+2)

        do i = 0,N+2
                X(i) = i*dx
                if(X(i).LT.diaph) then
                        u1(i) = rhol
                        u2(i) = rul
                        u3(i) = retl
                elseif(X(i).GE.diaph) then
                        u1(i) = rhor
                        u2(i) = rur
                        u3(i) = retr
                endif
        enddo

        end subroutine initial



        subroutine AUSM(gamma,rl,rul,retl,rr,rur,retr,f1,f2,f3)
        implicit none
        real(8) :: gamma, rl, rul, retl, rr, rur, retr, f1, f2, f3
        real(8) :: rhol, rhor, ul, ur, pl, pr, hl, hr, al, ar, Ml, Mr
        real(8) :: Mp, pp, Mm, pm, Mpm

!!! Convert conservative variables to primitive variables.
        rhol = rl
        rhor = rr
        ul = rul/rhol
        ur = rur/rhor
        pl = (gamma-1.0d0)*(retl - 0.5d0*rul*rul/rhol)
        pr = (gamma-1.0d0)*(retr - 0.5d0*rur*rur/rhor)
        hl = (retl+pl)/rhol
        hr = (retr+pr)/rhor
        al = sqrt(gamma*pl/rhol)
        ar = sqrt(gamma*pr/rhor)
        Ml = ul/al
        Mr = ur/ar

!!! Compute positive splitting of M and p.
        if(Ml.LE.-1.0d0) then
                Mp = 0.0d0
                pp = 0.0d0
```

```
    elseif(Ml.lt.1.0d0) then
        Mp = 0.25d0*(Ml+1.0d0)*(Ml+1.0d0)
        pp = 0.5d0*(1.0d0+Ml)*pl            !Choice One
!       pp = 0.25*pl*(1.0+Ml)*(1.0+Ml)*(2.0-Ml)     !Choice Two
    else
        Mp = Ml
        pp = pl
    endif

    !!! Compute negative splitting of M and p.
    if(Mr.LE.-1.0d0) then
        Mm = Mr
        pm = pr
    elseif(Mr.LT.1.0d0) then
        Mm = -0.25d0*(Mr-1.0d0)*(Mr-1.0d0)
        pm = 0.5d0*(1.0d0-Mr)*pr            !Choice One
!       pm = 0.25*pr*(1.0-Mr)*(1.0-Mr)*(2.0+Mr)     !Choice Two
    else
        Mm = 0.0d0
        pm = 0.0d0
    endif

    Mpm = Mp + Mm

    !!! Compute conserative numerical fluxes.
    !!! Splitting the flux vector F into a convective component F^(c) and a pressure component F^(p)
    f1 = MAX(0.0d0,Mpm)*rhol*al + MIN(0.0d0,Mpm)*rhor*ar
    f2 = MAX(0.0d0,Mpm)*rul*al + MIN(0.0d0,Mpm)*rur*ar + pp + pm
    f3 = MAX(0.0d0,Mpm)*rhol*hl*al + MIN(0.0d0,Mpm)*rhor*hr*ar

    return
    end subroutine AUSM
```