MATH5350 - Homework2

Luo Li

20155461

1 Question

Solve the linearied dynamic equation

$$\begin{pmatrix} \rho \\ u \end{pmatrix}_t + \begin{pmatrix} 0 & \rho_0 \\ \frac{a^2}{\rho_0} & 0 \end{pmatrix} \begin{pmatrix} \rho \\ u \end{pmatrix}_x = 0,$$

by discrete scheme

$$\mathbf{u}_j^{n+1} = \mathbf{u}_j^n + \frac{1}{\Delta x} \int_{t_n}^{t_{n+1}} (\mathbf{f}_{j-\frac{1}{2}} - \mathbf{f}_{j+\frac{1}{2}}) dt, \tag{1}$$

where $\mathbf{u} = \begin{pmatrix} \rho \\ u \end{pmatrix}$ and $\mathbf{f} = A\mathbf{u} = \begin{pmatrix} 0 & \rho_0 \\ \frac{a^2}{\rho_0} & 0 \end{pmatrix} \begin{pmatrix} \rho \\ u \end{pmatrix}$.

2 Riemann solver

For the Finite-Volumn scheme (1), solve the local Riemann problem:

$$\mathbf{u}_{j-\frac{1}{2}} = L(\mathbf{u}_{j-1}, \mathbf{u}_j),$$
$$\mathbf{u}_{j+\frac{1}{2}} = L(\mathbf{u}_j, \mathbf{u}_{j+1}),$$

where $L(\mathbf{u}_l, \mathbf{u}_r) = A\mathbf{u}^*$,

$$\mathbf{u}^* = \begin{pmatrix} \rho^* \\ u^* \end{pmatrix} = \beta_1 \begin{pmatrix} \rho_0 \\ -a \end{pmatrix} + \alpha_2 \begin{pmatrix} \rho_0 \\ a \end{pmatrix},$$

$$\beta_1 = \frac{a\rho_r - \rho_0 u_r}{2a\rho_0},$$
$$\alpha_2 = \frac{a\rho_l + \rho_0 u_l}{2a\rho_0}.$$

3 Time stepping

A three-order Strong Stability-Preserving Runge-Kutta (SSP-RK) method is used.

$$\begin{aligned}
\mathbf{u}^{(1)} &= \mathbf{u}^n + \Delta t L(\mathbf{u}^n), \\
\mathbf{u}^{(2)} &= \frac{3}{4}\mathbf{u}^n + \frac{1}{4}\mathbf{u}^{(1)} + \frac{1}{4}\Delta t L(\mathbf{u}^{(1)}), \\
\mathbf{u}^{(3)} &= \frac{1}{3}\mathbf{u}^n + \frac{2}{3}\mathbf{u}^{(2)} + \frac{2}{3}\Delta t L(\mathbf{u}^{(2)}).
\end{aligned}$$

4 Boundary condition

- Open boundary condition. Supposed there is one ghost point, the characteristic variable on which is $\mathbf{w}_g = \begin{pmatrix} w_{g1} \\ w_{g2} \end{pmatrix}$. According to the transit condition $w_1 = w_1^{(0)}(x + at)$ and $w_2 = w_2^{(0)}(x - at)$, the characteristic variable on ghost point must equal to the characteristic variable on boundary

  $$w_{g1} = w_{b1}, \quad w_{g2} = w_{b2}.$$

  Since the characteristic vector set K and its inverse K^{-1} are constant, the open boundary condition can be written as

  $$\mathbf{u}_g = \mathbf{u}_b.$$

- Reflection boundary condition

  $$\begin{pmatrix} \rho_g \\ u_g \end{pmatrix} = \begin{pmatrix} \rho_b \\ -u_b \end{pmatrix}.$$

5 Results

Since the initial velocity $u^{(0)}$ is zero, the characteristic variable

$$w_1(x) = \frac{\rho(x + a, 0)}{2\rho_0} \text{ and } w_2(x) = \frac{\rho(x - a, 0)}{2\rho_0}$$

propagate in counter directions, and the solution

$$\begin{pmatrix} \rho \\ u \end{pmatrix} = \begin{pmatrix} \rho_0 w_1 + \rho_0 w_2 \\ -a w_1 + a w_2 \end{pmatrix}$$

is a superposition of these two waves. It can be observed that the amplitude of the solution is half of the initial waves due to the factor $\frac{1}{2}$ in both w_1 and w_2.

- For the square perturbation wave case, it can be seen from Figure 1 that the solution separates into two square waves propagating in counter directions.

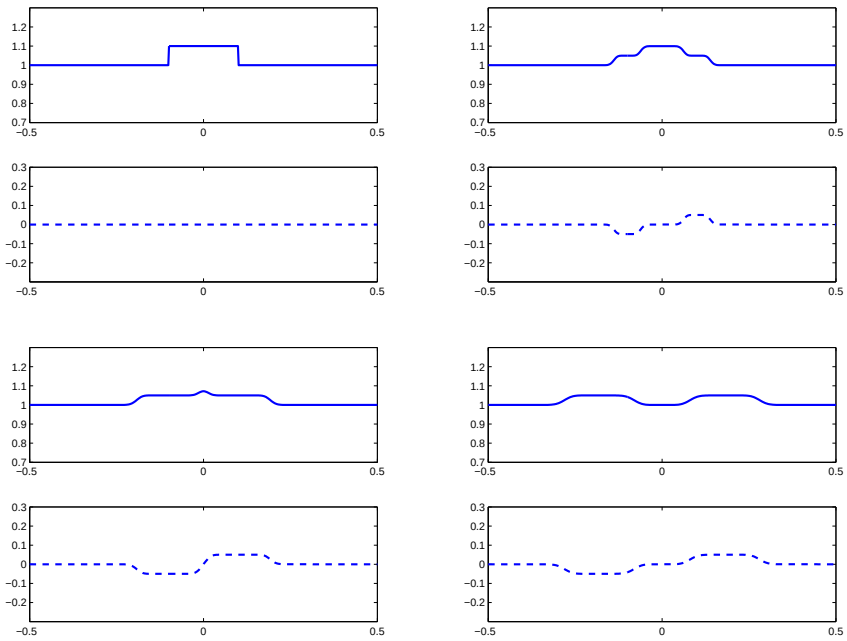

Figure 1: Square perturbation wave, $n = 500$, $C_{cfl} = 0.7$, final time $t_F = 1.0$, real line: ρ, dotted line: u, shown in every 8 time steps.

- For the triangle perturbation wave case, result appears similarly.
- For the sin wave with reflection boundary condition, the solution first separates into two distinct waves which propagate to the left and right boundary. After reflection, the two waves go back to interfere with each other.

6 Code

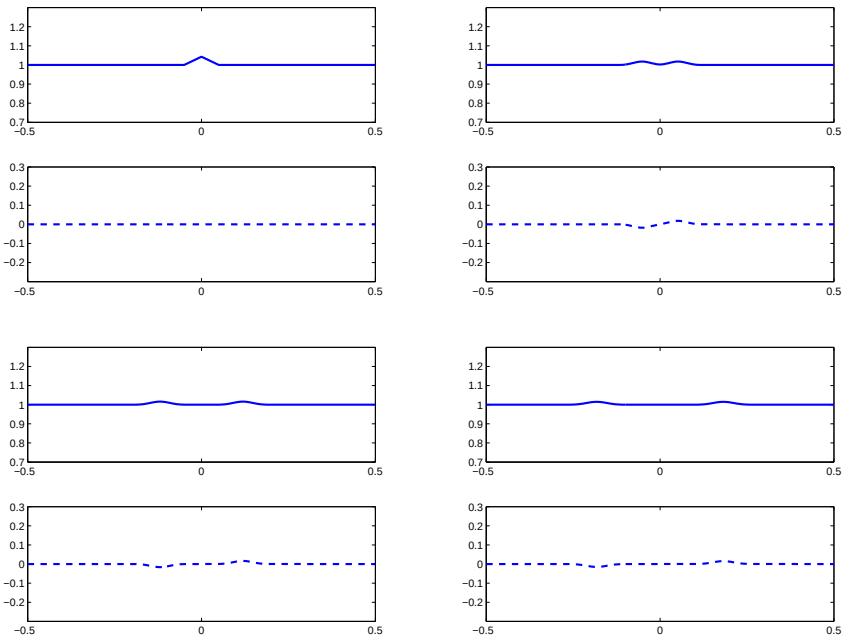

Figure 2: Triangle perturbation wave, $n = 500$, $C_{cfl} = 0.7$, final time $t_F = 1.0$, real line: ρ, dotted line: u, shown in every 8 time steps.

```
% Solve linear dynamic equation by exact Riemann solver and
    SSP-RK method
% n - number of spatial grid in one-dimension
% cfl - CFL number
% t_F - final time
% bc - boundary condition type: 1 - open boundary, 2 -
    reflection boundary
% flg - indicate number for different cases: 1 - square
    perturbation wave, 2 - triangle
% perturbation wave, 3 - sin wave
% called at MATLAB command line: linear_dynamic(500, 0.7,
    1.0, 1, 2)
% Author: Luo Li, lluoac@ust.hk
function linear_dynamic(n,cfl,t_F,bc,flg)
itmax=1000; % maximum of time steps
m=2; % number of ghost points
xmin=-0.5;
xmax=0.5;
dx=(xmax-xmin)/(n-1); % space step length
```

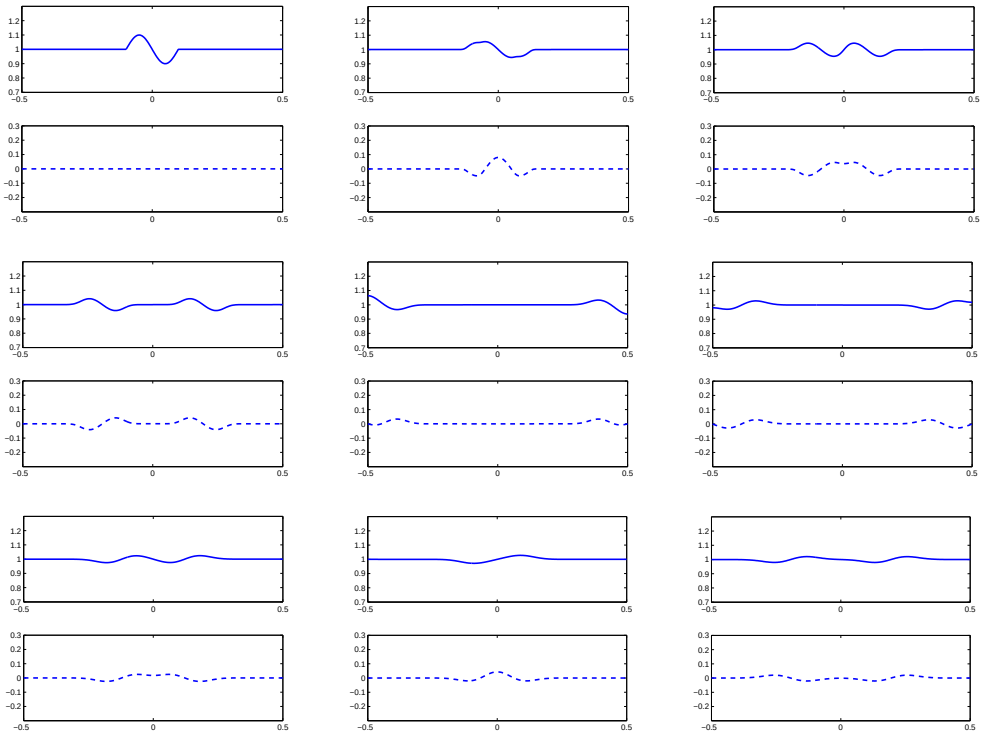

Figure 3: Sin wave with reflection boundary condition, $n = 500$, $C_{cfl} = 0.7$, final time $t_F = 1.0$, real line: ρ, dotted line: u, shown in every 5 time steps.

```
x=zeros(n,1);
u0=zeros(n+2*m,2);   % RK 0
u1=zeros(n+2*m,2);   % RK 1
u2=zeros(n+2*m,2);   % RK 2
u3=zeros(n+2*m,2);   % RK 3
% parameters
a=1.0;
% initialize
if flg==1 % square perturbation wave
    for i=1:n
        x(i)=-0.5+(i-1)*dx;
        if x(i)<-0.1
            u0(m+i,1)=0.0;
            u0(m+i,2)=0.0;
        elseif x(i)<=0.1
            u0(m+i,1)=0.1;
            u0(m+i,2)=0.0;
```

```
        else
            u0(m+i,1)=0.0;
            u0(m+i,2)=0.0;
        end
    end
elseif flg==2 % triangle perturbation wave
    for i=1:n
        x(i)=-0.5+(i-1)*dx;
        if x(i)<-0.05
            u0(m+i,1)=0.0;
            u0(m+i,2)=0.0;
        elseif x(i)<=0.0
            u0(m+i,1)=sqrt(3.0)/2.0*(x(i)+0.05);
            u0(m+i,2)=0.0;
        elseif x(i)<=0.05
            u0(m+i,1)=-sqrt(3.0)/2.0*(x(i)-0.05);
            u0(m+i,2)=0.0;
        else
            u0(m+i,1)=0.0;
            u0(m+i,2)=0.0;
        end
    end
elseif flg==3 % sin wave
    for i=1:n
        x(i)=-0.5+(i-1)*dx;
        if x(i)<-0.1
            u0(m+i,1)=0.0;
            u0(m+i,2)=0.0;
        elseif x(i)<=0.1
            u0(m+i,1)=0.1*sin(pi*(10.0*x(i)+1.0));
            u0(m+i,2)=0.0;
        else
            u0(m+i,1)=0.0;
            u0(m+i,2)=0.0;
        end
    end
end
if bc==1
% open boundary
for i=1:m
    u0(i,:)=u0(m+1,:);
    u0(m+n+i,:)=u0(m+n,:);
end
```

```
elseif bc==2
% reflection boundary
for i=1:m
    % left
    u0(i,1)=u0(2*m+1-i,1);  % density not change
    u0(i,2)=-u0(2*m+1-i,2); % velocity changes
    % right
    u0(m+n+i,1)=u0(m+n+1-i,1);
    u0(m+n+i,2)=-u0(m+n+1-i,2);
end
end
subplot(2,1,1)
plot(x,1.0+u0(m+1:m+n,1),'LineWidth',2);
axis([-0.5 0.5 0.7 1.3]); % called after plot()
subplot(2,1,2)
plot(x,u0(m+1:m+n,2),'LineWidth',2);
axis([-0.5 0.5 -0.3 0.3]); % called after plot()

% start time stepping
t=0.0;
for it=1:itmax
    % compute time step length dt=cfl*dx/|a|
    dt=cfl*dx/abs(a);
    % final time step length
    if t<t_F && t+dt>t_F
        dt=t_F-t;
    end
    t=t+dt
    pause;
    % if time is up, stop time stepping
    if t+dt>t_F || it==itmax
        break;
        subplot(2,1,1)
        plot(x,1.0+u0(m+1:m+n,1),'LineWidth',2);
        axis([-0.5 0.5 0.7 1.3]); % called after plot()
        subplot(2,1,2)
        plot(x,u0(m+1:m+n,2),'LineWidth',2);
        axis([-0.5 0.5 -0.3 0.3]); % called after plot()
    end

    lambda=dt/dx;
    % RK step 1
    for i=1:n
```

```
    u1(m+i,:)=u0(m+i,:)-lambda*(Riemann(u0(m+i,:),u0(m+
        i+1,:))-Riemann(u0(m+i-1,:),u0(m+i,:)));
end
if bc==1
% open boundary
for i=1:m
    u1(i,:)=u1(m+1,:);
    u1(m+n+i,:)=u1(m+n,:);
end
elseif bc==2
% reflection boundary
for i=1:m
    u1(i,1)=u1(2*m+1-i,1);
    u1(i,2)=-u1(2*m+1-i,2);
    u1(m+n+i,1)=u1(m+n+1-i,1);
    u1(m+n+i,2)=-u1(m+n+1-i,2);
end
end
% RK step 2
for i=1:n
    u2(m+i,:)=3.0/4.0*u0(m+i,:)+1.0/4.0*(u1(m+i,:)-
        lambda*(Riemann(u1(m+i,:),u1(m+i+1,:))-Riemann(
        u1(m+i-1,:),u1(m+i,:))));
end
if bc==1
% open boundary
for i=1:m
    u2(i,:)=u2(m+1,:);
    u2(m+n+i,:)=u2(m+n,:);
end
elseif bc==2
% reflection boundary
for i=1:m
    u2(i,1)=u2(2*m+1-i,1);
    u2(i,2)=-u2(2*m+1-i,2);
    u2(m+n+i,1)=u2(m+n+1-i,1);
    u2(m+n+i,2)=-u2(m+n+1-i,2);
end
end
% RK step 3
for i=1:n
    u3(m+i,:)=1.0/3.0*u0(m+i,:)+2.0/3.0*(u2(m+i,:)-
        lambda*(Riemann(u2(m+i,:),u2(m+i+1,:))-Riemann(
```

```
            u2(m+i-1,:),u2(m+i,:))));
    end
    if bc==1
    % open boundary
    for i=1:m
        u3(i,:)=u3(m+1,:);
        u3(m+n+i,:)=u3(m+n,:);
    end
    elseif bc==2
    % reflection boundary
    for i=1:m
        u3(i,1)=u3(2*m+1-i,1);
        u3(i,2)=-u3(2*m+1-i,2);
        u3(m+n+i,1)=u3(m+n+1-i,1);
        u3(m+n+i,2)=-u3(m+n+1-i,2);
    end
    end

    % swap
    u0(:,:)=u3(:,:);
    if bc==1
    % open boundary
    for i=1:m
        u0(i,:)=u0(m+1,:);
        u0(m+n+i,:)=u0(m+n,:);
    end
    elseif bc==2
    % reflection boundary
    for i=1:m
        u0(i,1)=u0(2*m+1-i,1);
        u0(i,2)=-u0(2*m+1-i,2);
        u0(m+n+i,1)=u0(m+n+1-i,1);
        u0(m+n+i,2)=-u0(m+n+1-i,2);
    end
    end
    subplot(2,1,1)
    plot(x,1.0+u0(m+1:m+n,1),'LineWidth',2);
    axis([-0.5 0.5 0.7 1.3]); % called after plot()
    subplot(2,1,2)
    plot(x,u0(m+1:m+n,2),'LineWidth',2);
    axis([-0.5 0.5 -0.3 0.3]); % called after plot()
end
```

```
% Riemann solver for linear dynamic equation
% ul - 1x2 array, state on the left
% ur - 1x2 array, state on the right
% flux - 1x2 array, output flux
% Author Luo Li, lluoac@ust.hk
function [flux]=Riemann(ul,ur)
% parameters
a=1.0;
rho0=1.0;
flux=zeros(1,2);
alpha2=(a*ul(1)+rho0*ul(2))/(2.0*a*rho0);
beta1=(a*ur(1)-rho0*ur(2))/(2.0*a*rho0);
u1=beta1*rho0+alpha2*rho0;
u2=beta1*(-a)+alpha2*a;
flux(1,1)=rho0*u2;
flux(1,2)=a*a/rho0*u1;
```