Num. #2: Hyperbolic PDE equation : transport equation - Correction

The programs are written with the MATLAB software.
For the exercise, the following functions are needed

- **Upwind method** :

```
% T is the final time, dt the time step
% L is the length of the interval, dx the space step
% uinit is the initial value (column vector),
% a is the velocity of the transport equation
% Upwind method
% Periodic boundary conditions
function[ufinal]=upwind(T,dt,L,dx,uinit,a)
    % Time discretization
    time=0:dt:T;
    Nt=length(time);
  % Space discretization COLUMN vector
    space=(0:dx:L)';
  % Initial datum - We calculate on N-1 points
  u=uinit(1:end-1);
  % Computation of the velocity at middle points velmid(i)=a(x_{i+1/2})
      velmid=a((space(1:end-1)+space(2:end))/2);
  % Computation of the vector  velmidm(i)=a(x_{i-1/2})
       velmidm=[velmid(end);velmid(1:end-1)];
  % upwind method
  for i=1:Nt
      % Periodic boundary conditions
      % Computation of vectors up(i)=u_{i+1}, um(i)=u_(i-1)
      up=[u(2:end);u(1)];
      um=[u(end);u(1:end-1)];
      % computation of flux
      Fp=(velmid.*(u+up)-abs(velmid).*(up-u))/2;
      Fm=(velmidm.*(um+u)-abs(velmidm).*(u-um))/2;
      u=u-dt/dx*(Fp-Fm);
  end
  ufinal=[u;u(1)];
```

- **Lax-Friedrichs method** :

```
% Lax Friedrichs method
```

```
% Periodic boundary conditions
function[ufinal]=LaxFriedrichs(T,dt,L,dx,uinit,a)
    % Time discretization
    time=0:dt:T;
    Nt=length(time);
    % Space discretization COLUMN vector
    space=(0:dx:L)';
 % Initial datum - We calculate on N-1 points
    u=uinit(1:end-1);
  % Computation of the velocity vel(i)=a(x_{i})
      vel=a(space(1:end-1));
    % Computations of vectors velp(i)=a(x_{i+1}) and velm(i)=a(x_{i-1})
      velp=[vel(2:end);vel(1)];
      velm=[vel(end);vel(1:end-1)];
  % Lax-Friedrichs method
  for i=1:Nt
      % Periodic boundary conditions
      % Computation of vectors up(i)=u_{i+1}, um(i)=u_(i-1)
      up=[u(2:end);u(1)];
      um=[u(end);u(1:end-1)];
      % computation of flux
      Fp=(vel.*u+velp.*up)/2-dx*(up-u)/2/dt;
      Fm=(velm.*um+vel.*u)/2-dx*(u-um)/2/dt;
      u=u-dt/dx*(Fp-Fm);
  end
  ufinal=[u;u(1)];
```

- **Lax-Wendroff method** :

```
% Lax Wendroff method
% Periodic boundary conditions
function[ufinal]=LaxWendroff(T,dt,L,dx,uinit,a)
   % Time discretization
   time=0:dt:T;
   Nt=length(time);
  % Space discretization COLUMN vector
  space=(0:dx:L)';
  % Initial datum - We calculate on N-1 points
   u=uinit(1:end-1);
  % Computation of the velocity and the velocity at middle points
   vel=a(space(1:end-1));
```

```
 velmid=a((space(1:end-1)+space(2:end))/2);
 % Computations of vectors velp(i)=a(x_{i+1}) and velm(i)=a(x_{i-1})
 velp=[vel(2:end);vel(1)];
 velm=[vel(end);vel(1:end-1)];
% Computation of the vector  velmidm(i)=a(x_{i-1/2})
velmidm=[velmid(end);velmid(1:end-1)];
% Lax-Wendroff method
 for i=1:Nt
    % Periodic boundary conditions
    % Computation of vectors up(i)=u_{i+1}, um(i)=u_(i-1)
    up=[u(2:end);u(1)];
    um=[u(end);u(1:end-1)];
    % computation of flux
    Fp=(vel.*u+velp.*up)/2-dt*velmid.*(velp.*up-vel.*u)/2/dx;
    Fm=(velm.*um+vel.*u)/2-dt*velmidm.*(vel.*u-velm.*um)/2/dx;
    u=u-dt/dx*(Fp-Fm);
end
ufinal=[u;u(1)];
```

Exercise

1. To begin with, define a vector with the discrete points (x_i) which discretize the interval $[0,5]$ with a space step $\Delta x = 0.1$. Define a discretization of the three following initial data and plot them :

$$u_0(x) = \mathrm{e}^{-(x-2)^2/0.1} \tag{1a}$$

$$u_0(x) = \begin{cases} 1-|x-2| & \text{if } 1 \le x \le 3 \\ 0 & \text{otherwise} \end{cases} \tag{1b}$$

$$u_0(x) = \begin{cases} 1 & \text{if } 1 \le x \le 2 \\ 0 & \text{otherwise} \end{cases} \tag{1c}$$

```
 % Space discretization
L=5;
dx=0.1;
space=(0:dx:L)';
%% space should be a column vector
% Initial datum 1
uinit=exp(-(space-2).^2/0.1);
% Initial datum 2
space1=space(space<1);
space2=space((space>=1)&(space<=3));
```

```
space3=space(space>3);
uinit=[zeros(size(space1));1-abs(space2-2); zeros(size(space3))];
% Initial datum 3
space1=space(space<1);
space2=space((space>=1)&(space<=2));
space3=space(space>2);
uinit=[zeros(size(space1));ones(size(space2)); zeros(size(space3))];
% Plot of the initial datum
plot(space,uinit);
```

2. Implement the resolution of the following equations :

$$\partial_t u + \partial_x u = 0, \tag{2a}$$

$$\partial_t u + \partial_x (\sin(\frac{2\pi}{L} x) u) = 0, \tag{2b}$$

using the three methods presented above and a time step $\Delta t = 0.04$ until time $T = 1$. We still consider the interval $[0,5]$ with a space step $\Delta x = 0.1$ and we will use function (1a) as an initial datum. We take some periodic boundary conditions $u(t,0) = u(t,L)$.

```
% First example
% Clear variables and clear graphic
clear;
clf;
% Space discretization
L=5;
dx=0.1;
space=(0:dx:L)';
% Time discretization
T=1;
dt=0.04;
% Initial datum 1
uinit=exp(-(space-2).^2/0.1);
% Velocity of the transport equation -- a=1 or a=sin(2*pi*x/L)
a=inline('ones(size(x))');
%a=inline('sin(2*pi*x/5)');
% Approximated solution
uUp=upwind(T,dt,L,dx,uinit,a);
plot(space,uUp,'b');
hold on;
uLF=LaxFriedrichs(T,dt,L,dx,uinit,a);
plot(space,uLF,'r');
```

```
uLW=LaxWendroff(T,dt,L,dx,uinit,a);
plot(space,uLW,'g');
```

3. Use the previous program for equation (2a) with initial datum (1a) with the following time steps : $\Delta t = 0.2, 0.1, 0.09, 0.05$. What do you notice ?

```
clear;
clf;
% Space discretization
L=5;
dx=0.1;
% Time discretization
T=1;
dt=0.2;
%dt=0.1;
%dt=0.09;
% dt=0.05;
space=(0:dx:L)';
% Initial datum 1
uinit=exp(-(space-2).^2/0.1);
% Velocity of the transport equation -- a=1
a=inline('ones(size(x))');
% Approximated solution
uUp=upwind(T,dt,L,dx,uinit,a);
plot(space,uUp,'b');
hold on;
uLF=LaxFriedrichs(T,dt,L,dx,uinit,a);
plot(space,uLF,'r');
uLW=LaxWendroff(T,dt,L,dx,uinit,a);
plot(space,uLW,'g');
legend('upwind','Lax friedrichs','Lax Wendroff')
```

4. We still consider equation (2a) with initial datum (1a) and the following parameters $T = 1$, $L = 5$, $\Delta t = 0.95\Delta x$. Compare the order of the three methods by plotting a graph in a log-log scale, which represents the evolution of L^2 error with respect to the space step Δx.

```
clear;
clf;
% Space discretization
L=5;
```

```
% Different space steps
SpaceStep=[0.1, 0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001];
for k=1:length(SpaceStep),
dx=SpaceStep(k);
space=(0:dx:L)';
% Time discretization
T=1;
dt=0.95*dx;
time=0:dt:T;
Nt=length(time);
Tsimu=dt*Nt;
% Velocity of the transport equation -- a=1
a=inline('ones(size(x))');
% Initial datum 1
uinit=exp(-(space-2).^2/0.1);
% Exact solution 1
uexact=exp(-(space-2-Tsimu).^2/0.1);
% Approximated solution
uUp=upwind(T,dt,L,dx,uinit,a);
ErrorUp(k)=sqrt(dx)*norm(uUp-uexact);
uLF=LaxFriedrichs(T,dt,L,dx,uinit,a);
ErrorLF(k)=sqrt(dx)*norm(uLF-uexact);
uLW=LaxWendroff(T,dt,L,dx,uinit,a);
ErrorLW(k)=sqrt(dx)*norm(uLW-uexact);
end
% Clear the figure
clf;
% Graph of the errors
loglog(SpaceStep,ErrorUp)
hold on;
loglog(SpaceStep,ErrorLF,'r')
loglog(SpaceStep,ErrorLW,'g')
% Legend for the graph
legend('Upwind', 'Lax Friedrichs', 'Lax Wendroff');
```

5. Compare the three schemes for equation (2a) in the case of the two other initial data (1b) and (1c), with $T = 1$, $L = 5$, $\Delta x = 0.01$, $\Delta t = 0.95\Delta x$. . What do you notice ?

```
clear;
clf;
% Space discretization
```

```
L=5;
dx=0.01;
space=(0:dx:L)';
% Time discretization
T=1;
dt=0.95*dx;
% Initial datum 2
space1=space(space<1);
space2=space((space>=1)&(space<=3));
space3=space(space>3);
uinit=[zeros(size(space1));1-abs(space2-2); zeros(size(space3))];
% Initial datum 3
% space1=space(space<1);
% space2=space((space>=1)&(space<=2));
% space3=space(space>2);
% uinit=[zeros(size(space1));ones(size(space2)); zeros(size(space3))];
plot(space,uinit,'k');
% Velocity of the transport equation -- a=1
a=inline('ones(size(x))');
% Approximated solution
uUp=upwind(T,dt,L,dx,uinit,a);
plot(space,uUp);
hold on;
uLF=LaxFriedrichs(T,dt,L,dx,uinit,a);
plot(space,uLF,'r');
uLW=LaxWendroff(T,dt,L,dx,uinit,a);
plot(space,uLW,'g');
legend('initial','upwind','Lax friedrichs','Lax Wendroff')
```