Exercise set 4

Numerical Methods for Hyperbolic Partial Differential Equations
IMATH, FS-2020
Lecturer: Dr. Philipp Öffner
Teaching Assistant: Davide Torlo

Problem 4.1 Linear upwind and downwind schemes (6pts)

We study in this exercise the upwind and downwind methods for linear transport equation

$$\partial_t u(t,x) + c\partial_x u(t,x) = 0, \tag{1}$$

with $c \in \mathbb{R}$ $x \in [a,b]$ and $t \in [0,T]$.

The two methods consider a first order finite difference discretization which follows the flux or goes in the opposite verse. We can define the upwind scheme as

$$u_j^{n+1} = u_j^n - \frac{\Delta t c^+}{\Delta x}\left(u_j^n - u_{j-1}^n\right) - \frac{\Delta t c^-}{\Delta x}\left(u_{j+1}^n - u_j^n\right), \tag{2}$$

while the downwind scheme is defined as

$$u_j^{n+1} = u_j^n - \frac{\Delta t c^-}{\Delta x}\left(u_j^n - u_{j-1}^n\right) - \frac{\Delta t c^+}{\Delta x}\left(u_{j+1}^n - u_j^n\right), \tag{3}$$

where $c^+ = \max(c,0)$ and $c^- = \min(c,0)$.

1. Prove that the previous scheme can also be written in the more general form

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{c(u_{j+1}^n - u_{j-1}^n)}{2\Delta x} = \frac{\sigma|c|(u_{j+1}^n - 2u_j^n + u_{j-1}^n)}{2\Delta x}, \quad \text{with } \sigma = \begin{cases} +1, & \text{(upwind)}, \\ -1, & \text{(downwind)}. \end{cases} \tag{4}$$

   Solution

   Consider the upwind scheme of formulation (4):

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} + \frac{c(u_{j+1}^n - u_{j-1}^n)}{2\Delta x} - \frac{|c|(u_{j+1}^n - 2u_j^n + u_{j-1}^n)}{2\Delta x} = 0 \tag{5}$$

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\left(\frac{c-|c|}{2}u_{j+1}^n - \frac{c+|c|}{2}u_{j-1}^n + |c|u_j^n + \frac{c-c}{2}u_j^n\right) \tag{6}$$

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\left(\frac{c-|c|}{2}(u_{j+1}^n - u_j^n) - \frac{c+|c|}{2}(u_{j-1}^n - u_j^n)\right) \tag{7}$$

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\left(c^-(u_{j+1}^n - u_j^n) + c^+(u_j^n - u_{j-1}^n)\right), \tag{8}$$

   where we have used the relations $c^+ = \frac{|c|+c}{2}$ and $c^- = \frac{c-|c|}{2}$.

2. Compute the von Neumann analysis of the downwind scheme (3) for $c > 0$ and say if the scheme is stable or not.

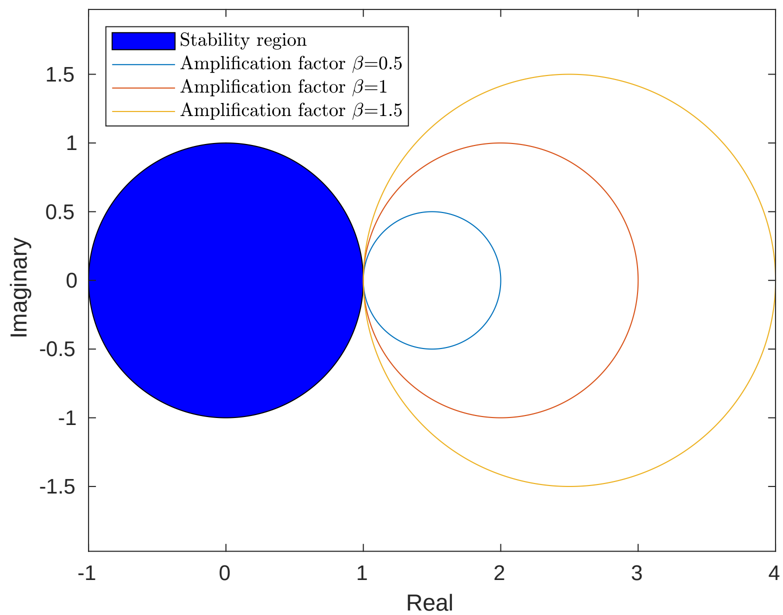

Figure 1: Amplification factor for the downwind method

Solution

For $c > 0$ the downwind scheme reads

$$u_j^{n+1} = u_j^n - \frac{\Delta t c^+}{\Delta x}\left(u_{j+1}^n - u_j^n\right), \tag{9}$$

if we define $\beta := \frac{\Delta t c^+}{\Delta x} > 0$, the von Neumann analysis for any wave number $k \in \mathbb{N}$ results in

$$e^{\alpha t^{n+1}} e^{ikx_j} = e^{\alpha t^n} e^{ikx_j} - \beta\left(e^{\alpha t^n} e^{ikx_{j+1}} - e^{\alpha t^n} e^{ikx_j}\right) \tag{10}$$

$$e^{\alpha \Delta t} = 1 - \beta\left(e^{ik\Delta x} - 1\right). \tag{11}$$

If we plot this number in the complex plain for any possible value of $k\Delta x$, we see that it describe a circumference with center in $1+\beta$ and radius β, see Figure 1. Hence, it never intersect the stability region $|e^{\alpha \Delta t}| < 1$.

3. Prove that the upwind scheme is total variation diminishing (use periodic boundary conditions), i.e.,

$$TV(\mathbf{u}^{n+1}) \leq TV(\mathbf{u}^n), \quad \text{with } TV(\mathbf{u}) = \sum_j \Delta x |u_{j+1} - u_j|. \tag{12}$$

Solution

Let us compute the total variation of the solution at time t^{n+1}, with the usual notation $a = x_0 \equiv x_N = b$ and $x_{-1} = x_{N-1}$:

$$\sum_{j=1}^{N} |u_j^{n+1} - u_{j-1}^{n+1}| = \sum_j \Bigg| (u_j^n - u_{j-1}^n) - \frac{\Delta t c^+}{\Delta x}\left(u_j^n - u_{j-1}^n\right) - \frac{\Delta t c^-}{\Delta x}\left(u_{j+1}^n - u_j^n\right) \tag{13}$$

$$+ \frac{\Delta t c^+}{\Delta x}\left(u_{j-1}^n - u_{j-2}^n\right) + \frac{\Delta t c^-}{\Delta x}\left(u_j^n - u_{j-1}^n\right) \Bigg|. \tag{14}$$

Now, we collect the difference $\left(u_j^n - u_{j-1}^n\right)$ and we get

$$\sum_j \left| \left(1 - \frac{\Delta t}{\Delta x}(c^+ - c^-)\right)(u_j^n - u_{j-1}^n) - \frac{\Delta t c^-}{\Delta x}\left(u_{j+1}^n - u_j^n\right) + \frac{\Delta t c^+}{\Delta x}\left(u_{j-1}^n - u_{j-2}^n\right) \right|. \tag{15}$$

Now, we notice that $-\frac{\Delta t c^-}{\Delta t}$ and $\frac{\Delta t c^+}{\Delta t}$ are positive and, if we apply the CFL conditions

$$1 \geq \frac{\Delta t}{\Delta x}(c^+ - c^-) = \frac{|c|\Delta t}{\Delta x}, \tag{16}$$

also the first coefficient is positive. Moreover, the sum of the 3 coefficients is equal to 1. Using the fact that the absolute value is a convex function, we have that

$$\sum_{j=1}^N |u_j^{n+1} - u_{j-1}^{n+1}| \leq \sum_j \left(1 - \frac{\Delta t}{\Delta x}(c^+ - c^-)\right)|u_j^n - u_{j-1}^n| - \frac{\Delta t c^-}{\Delta x}|u_{j+1}^n - u_j^n| + \frac{\Delta t c^+}{\Delta x}|u_{j-1}^n - u_{j-2}^n|. \tag{17}$$

Now, rearranging the terms of the sum and using the periodic conditions we get

$$\sum_{j=1}^N |u_j^{n+1} - u_{j-1}^{n+1}| \leq \sum_j \left(1 - \frac{\Delta t}{\Delta x}(c^+ - c^-)\right)|u_j^n - u_{j-1}^n| - \frac{\Delta t c^-}{\Delta x}|u_j^n - u_{j-1}^n| + \frac{\Delta t c^+}{\Delta x}|u_j^n - u_{j-1}^n| \tag{18}$$

$$\frac{TV(\mathbf{u}^{n+1})}{\Delta x} \leq \sum_j |u_j^n - u_{j-1}^n| = \frac{TV(\mathbf{u}^n)}{\Delta x}. \tag{19}$$

4. Code both schemes in the form (4) with periodic boundary conditions, in a function where inputs are c the speed of the transport equation, N number of subintervals of the domain, CFL number ($\Delta t := \text{CFL}\Delta x/|c|$), T the final time, a and b the domain extrema, u_0 the initial conditions.

Solution

```
function [u,t,x, varargout] = updownwind(c, N, cfl, T, a,b, u0, sigma, varargin)
%LAXFR   Solve the advection equation u_t + c u_x = 0 using the
%   upwind/downwind scheme. The solution is computed on the periodic domain
%   [-a, b) up to t=tMax.

    if nargin>8
        with_error=1;
    else
        with_error=0;
    end

  % Create the mesh
  x = linspace(a, b, N+1);
  dx = x(2)-x(1);
  x = [x(N), x];     % Mesh with ghost cells {x_0, ..., x_{N+1}}
  j = 2 : N+1;     % Index into the internal mesh  1,...,N

  % Set initial data and advection speed
  aMax = abs(c);
    if with_error
        u_exact=@(x,t) u0(x-c*t);
      uu_exact(1,:) = u_exact(x,0);
    end
    u(1,:) = u0(x);


  % Set the timestep and make sure that tMax/dt is an integer
```

```
  dt = cfl * dx/aMax;
  nt = ceil(T / dt)+1;
  t = linspace(0, T, nt);
  dt = t(2)-t(1);
  lambda = dt/dx;

  % Run the simulation
  for n = 2:nt
    % Update the solution
    u(n,j) = u(n-1,j) - lambda/2*c*(u(n-1,j+1)-u(n-1,j-1)) ...
            + sigma*abs(c)*lambda/2*(u(n-1,j+1)-2*u(n-1,j)+u(n-1,j-1));

    % Set boundary conditions
    u(n,1) = u(n,N+1);
    u(n,N+2) = u(n,2);
        if with_error
            uu_exact(n,:) = u_exact(x,t(n));
        end
    end
    if with_error
      errL2end= norm(u(end,:)-uu_exact(end,:))*sqrt(dx);
        varargout={uu_exact,errL2end};
    else
        varargout={};
    end
  % Remove the ghost cell to the left
  x = x(2:end-1);
  u = u(:,2:end-1);
end
```

Listing 1: updownwind.m

5. Use the coefficients $c = 1$, $N = 200$, $\text{CFL} = 0.9$, $T = 1$, $a = -1$, $b = 1$, $u_0 = \cos(\pi x)$ to test the schemes. Plot the total variation of the numerical solutions with respect to time in an appropriate scale. What do you observe?

   Solution

   The experiments show that the downwind is unstable, while the upwind is stable. Check code 2. Indeed, in Figure 3 (left), we can see that the TV of the downwind scheme is exploding with time, while the upwind scheme is diminishing its TV.

6. Test the convergence order of the upwind scheme (2). For number of cells $N \in \{2^k | k = 1, \dots, 10\}$, run the upwind scheme and compute the final $\mathbb{L}^2$ error with respect to the exact solution

$$\|u(T) - u_{ex}(T)\|_2 := \sqrt{\Delta x \sum_{j=1}^{N} \left(u(T, x_j) - u_{ex}(T, x_j)\right)^2}. \tag{20}$$

   Plot the error vs Δx and a reference first order decay. Verify that they have the same rate of convergence.

   Solution

   The error of the upwind scheme is decreasing with first order accuracy, as expected. See Figure 3 (right) and code 2.

```
%% run downwind scheme

close all
plot_evolution = true;
```

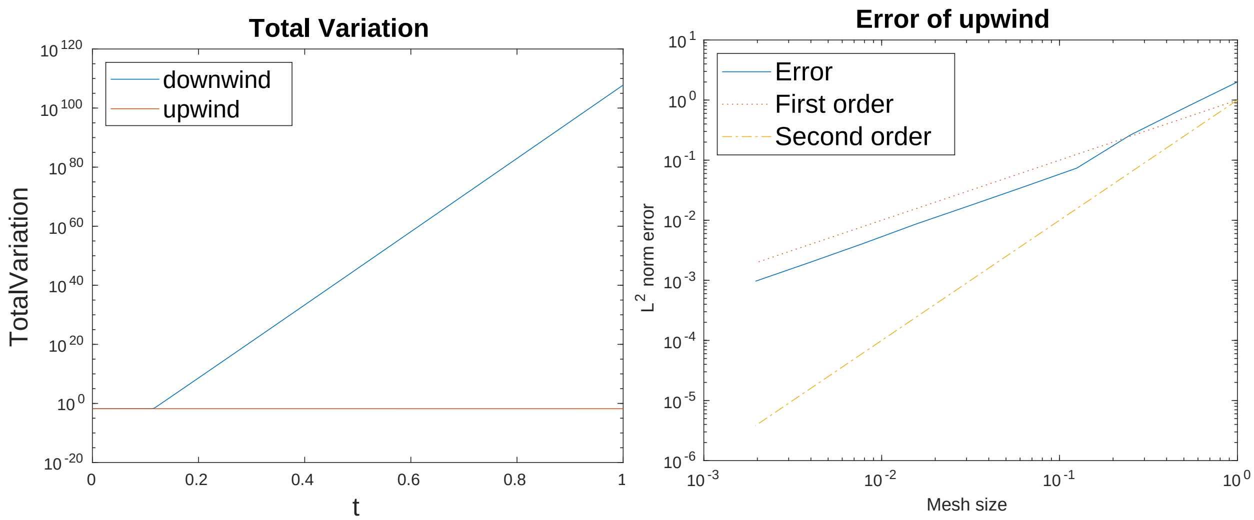

Figure 2: Total variation of upwind and downwind and error of upwind

```
%Problem data
a=-1;
b=1;
u0=@(x) cos(pi*x);
T=1;
c=1;

%%
% run downwind scheme
sigma = -1;
Nx = 500;
tfin = 1.;
cfl = 0.9;
%
with_error=1;
[udown,t,x, ex] = updownwind(c, Nx, cfl, tfin, a,b, u0,sigma, with_error);

%% run upwind scheme
%
sigma = +1;
Nx = 500;
tfin = 1.;
cfl = 0.9;
%
[uup,t,x] = updownwind(c, Nx, cfl, tfin, a,b, u0,sigma);

%% plot evolution
dx = x(2)-x(1);
u0 = @(x) cos(pi*x);

if(plot_evolution)
    for n=[1:ceil(numel(t)/20):numel(t),numel(t)]
        %
        figure(1)
        %
        subplot(211)
        plot(x,udown(n,:),...
            x,u0(x-t(n)),'r--')
        legend('numerical','analytical','Location','SE')
```

```
        xlabel x
        ylabel u
        title('downwind')
        ylim([-1.5,1.5])

        %
        subplot(212)
        plot(x,uup(n,:),...
            x,u0(x-t(n)),'r--')
        legend('numerical','analytical','Location','SE')
        xlabel x
        ylabel u
        title(sprintf('upwind, t=%g',cfl,t(n)))
        ylim([-1.5,1.5])

        %
        drawnow
        pause(.02)
    end
end

%% plot TV instability for upwind

fig=figure
TVup=zeros(size(t));
TVdown=zeros(size(t));
udown_1=zeros(size(udown));
udown_1(:,1:end-1)=udown(:,2:end);
udown_1(:,end)=udown(:,1);
uup_1=zeros(size(uup));
uup_1(:,1:end-1)=uup(:,2:end);
uup_1(:,end)=uup(:,1);
for n=1:numel(t)
    TVdown(n)= sum(abs(udown(n,:)-udown_1(n,:)))*dx;
    TVup(n)= sum(abs(uup(n,:)-uup_1(n,:)))*dx;
    %
end

semilogy(t,TVdown,'DisplayName','downwind')
hold on
semilogy(t,TVup,'DisplayName','upwind')

xlabel('t','FontSize',16)
ylabel('TotalVariation','FontSize',16)
title('Total Variation','FontSize',16)
legend('FontSize',15,'Location','NW')
hold off
saveas(fig,'TVwinds.pdf')

%% plot error convergence
tfin = 1.;
cfl = 0.9;

with_error=1;
nn=10;
Nxs=2.^[1:nn];
errup=zeros(size(Nxs));
for k=1:nn
    Nx=Nxs(k);
    hs(k)=2/Nx;
    [uup,t,x,ex, err] = updownwind(c, Nx, cfl, tfin, a,b, u0, 1  ,with_error);
    errup(k)=err;
end

```

```
fig=figure()
loglog(hs,errup,'DisplayName', 'Error')
title('Error of upwind','FontSize',16)
ylabel('L^2 norm error')
xlabel('Mesh size')
hold on
loglog(hs,hs,':','DisplayName','First order')
loglog(hs,hs.^2,'-.','DisplayName','Second order')
legend('Location','NW','FontSize',16)
saveas(fig,'errorUpwind.pdf')
```

Listing 2: numerical_experiments.m

Problem 4.2 Conservative upwind (4pts)

For the nonlinear equations, the definition of upwind scheme is not unique as there is no unique weak solution. Consider the Burgers' equation in two forms: the conservative

$$\partial_t u(t,x) + \partial_x \left(\frac{u(t,x)^2}{2} \right) = 0 \tag{21}$$

and the quasi linear form

$$\partial_t u(t,x) + u(t,x)\partial_x u(x,t) = 0. \tag{22}$$

We consider only nonnegative functions u for this exercise, hence, the wind blows from left to right. The two formulations lead to two different upwind schemes. Consider the conservative upwind scheme

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x} \frac{(u_j^n)^2 - (u_{j-1}^n)^2}{2} \tag{23}$$

and the quasi linear upwind given by

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x} u_j^n \left(u_j^n - u_{j-1}^n \right). \tag{24}$$

1. Prove that (23) is conservative and that (24) is not conservative.

   Solution

   The scheme (23) is conservative because it can be put in the conservation form through the numerical flux definition $f_{j+1/2} = (u_j)^2/2$. Hence, the scheme (23) can be written as

   $$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}(f_{j+1/2}^n - f_{j-1/2}^n). \tag{25}$$

   (22) is not conservative. Take as counterexample the vector $\mathbf{u}^n = (0,1,0,0,\dots,0)$. If we compute

   $$\sum_j u_j^{n+1} = \sum_j u_j^n - \frac{\Delta t}{\Delta x} \sum_j u_j^n (u_j^n - u_{j-1}^n) = \sum_j u_j^n - \frac{\Delta t}{\Delta x}. \tag{26}$$

2. Given the Riemann problem

   $$u_0(x) = \begin{cases} 1 & \text{if } x < 0, \\ 0 & \text{if } x \geq 0, \end{cases} \tag{27}$$

   write the exact solution for the conservation law (21). Find the solution that the method (24), derived by the quasi linear form, would give.

   Hint: take a space discretization and evolve for one time step the solution for *interesting* points with the scheme (24). What do you observe? How will the method evolve for the all the time steps?

Solution

Consider a space discretization $\{x_j\}_j = 1^N$, where $x_k < 0$ and $x_{k+1} \geq 0$ for a certain $k \in \{2, \dots, N-1\}$. The numerical solution at the time zero will be

$$u_j^0 = \begin{cases} 1 & \text{if } j \leq k, \\ 0 & \text{if } j > k. \end{cases} \tag{28}$$

For $j \leq k$ it is clear that $u_j^1 = 1$ as $u_j^0 - u_{j-1}^0 = 0$. On the other hand, the same reasoning applies also for $j > k+1$, hence $u_j^1 = 0$ for those j. For $j = k+1$, we have $u_{k+1}^1 = u_{k+1}^0 - u_{k+1}^0(u_{k+1}^0 - u_k^0) = 0 + 0 \cdot 1 = 0$. So,

$$u_j^1 = \begin{cases} 1 & \text{if } j \leq k, \\ 0 & \text{if } j > k. \end{cases} \tag{29}$$

By induction, one can prove that the scheme does not change the solution over time.

3. Code the two methods with Dirichlet boundary conditions on the left and outflow boundary conditions on the right (no conditions). Test the methods with the Riemann problem (27) and domain $[-1, 1]$, CFL=0.75, $T = 1$. Plot the two solutions and the exact one at final time $T = 1$.

4. Test the codes again first changing the Riemann problem into

$$u_0(x) = \begin{cases} 1.5 & \text{if } x < 0, \\ 0.3 & \text{if } x \geq 0. \end{cases} \tag{30}$$

What is the exact solution? Plot again the final solutions of the numerical methods and compare them with the exact one. And what happens if you choose the CFL=1.5?

Solution

```
function [u,x,t] = upwindBurgers(N, CFL, T, a, b, u0)

x=linspace(a,b,N+1);
dx=x(2)-x(1);
maxu=max(u0(x));
dt=dx*CFL/maxu;
Nt=ceil(T/dt);
dt=T/Nt;
t=linspace(0,T,Nt+1);

u=zeros(Nt+1,N+1);
u(1,:)=u0(x);
j=2:N+1;
for kt=2:Nt+1
    u(kt,j)=u(kt-1,j) - dt/dx*(u(kt-1,j).^2-u(kt-1,j-1).^2)/2;
    u(kt,1)=u0(x(1));
end

end
```

Listing 3: upwindBurgers.m

```
function [u,x,t] = upwindQuasiBurgers(N, CFL, T, a, b, u0)

x=linspace(a,b,N+1);
dx=x(2)-x(1);
maxu=max(u0(x));
dt=dx*CFL/maxu;
Nt=ceil(T/dt);
```

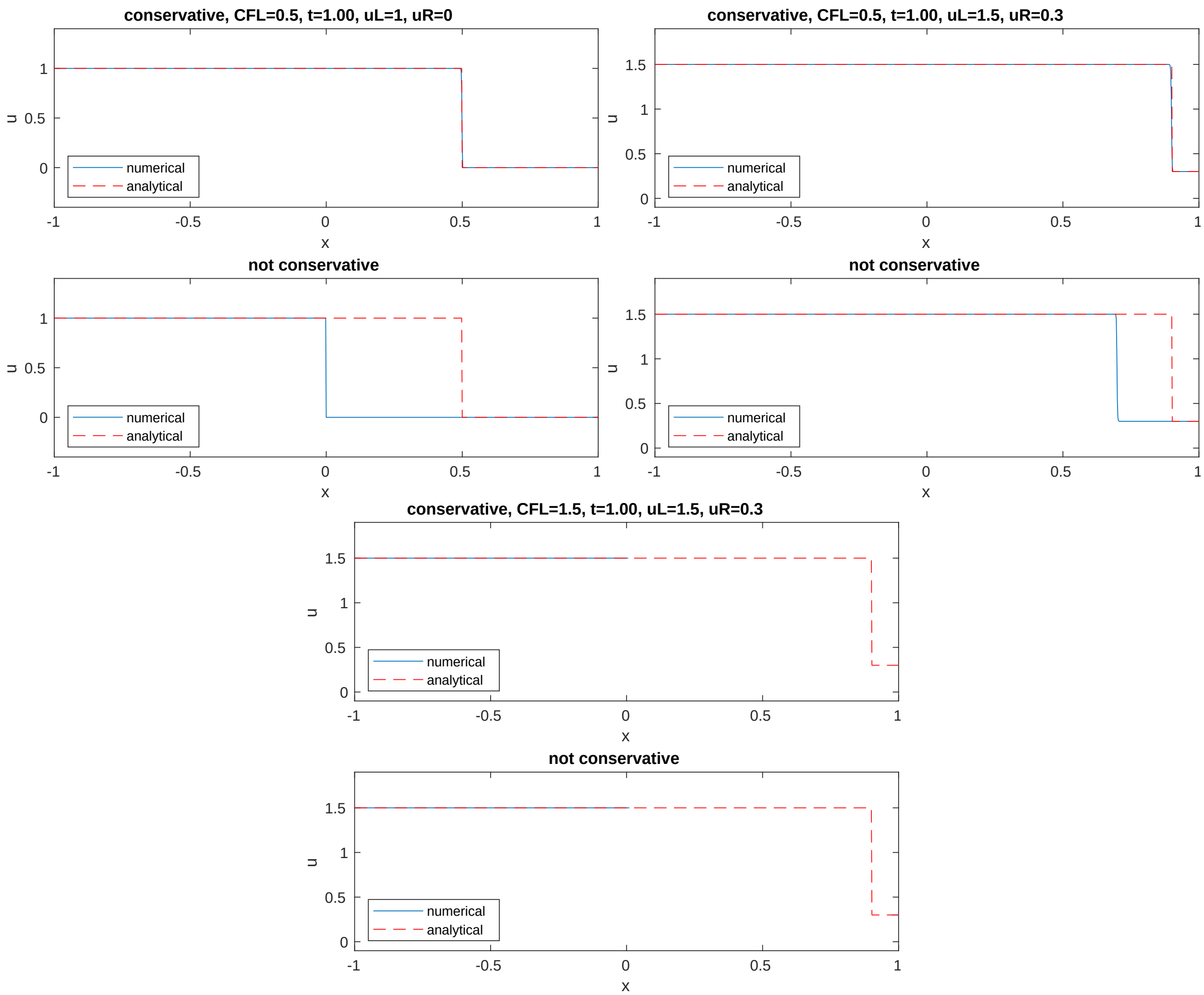

Figure 3: Conservative and non conservative upwind schemes on Burgers equation for different CFL and Riemann problems

```
dt=T/Nt;
t=linspace(0,T,Nt+1);

u=zeros(Nt+1,N+1);
u(1,:)=u0(x);
j=2:N+1;
for kt=2:Nt+1
    u(kt,2:end)=u(kt-1,j) - dt/dx*u(kt-1,j).*(u(kt-1,j)-u(kt-1,j-1));
    u(kt,1)=u0(x(1));
end

end
```

Listing 4: UpwindQuasiBurgers.m

```
% Test the Riemann problem on Burgers' equation

```

```matlab
close all
plot_evolution = true;

uL=1.5;
uR=0.3;

uminplot=min(uL,uR)-0.4;
umaxplot=max(uL,uR)+0.4;

f=@(u) u.^2/2;
u0=@(x) uL*(x<0)+uR*(x>=0);


%exact solution with RH conditions
s=(f(uR)-f(uL))/(uR-uL);
u_ex=@(x,t) u0(x-s*t);


N=1000;
T=1;
CFL=1.5;
a=-1;
b=1;

[uCons,x,t]=upwindBurgers(N,CFL,T,a,b,u0);
uNonCons=upwindQuasiBurgers(N,CFL,T,a,b,u0);

%%
fig=figure(1);
if(plot_evolution)
    for n=[1:ceil(numel(t)/20):numel(t),numel(t)]
        %
        figure(1)
        %
        subplot(211)
        plot(x,uCons(n,:),...
            x,u_ex(x,t(n)),'r--')
        legend('numerical','analytical','Location','SW')
        xlabel x
        ylabel u
        title(sprintf('conservative, CFL=%g, t=%1.2f, uL=%g, uR=%g',CFL,t(n),uL,uR))
        ylim([uminplot,umaxplot])

        %
        subplot(212)
        plot(x,uNonCons(n,:),...
            x,u_ex(x,t(n)),'r--')
        legend('numerical','analytical','forecast','Location','SW')
        xlabel x
        ylabel u
        title(sprintf('not conservative'))
        ylim([uminplot,umaxplot])

        %
        drawnow
        pause(.02)
    end
end

saveas(fig, 'Burgers2.pdf')
```

Listing 5: testBurgers.m

Submit the code for both exercises.

Organiser: Davide Torlo, Office: home (davide.torlo@math.uzh.ch)

Published: Mar 26, 2020

Due date: Apr 2, 2020, h10.00 (use the upload tool of my.math.uzh.ch, see wiki.math.uzh.ch/public/student_upload_homework or if you have troubles send me an email).