Project 5

Due on June 30th, 2017

Consider the transport equation:

$$\frac{\partial u}{\partial t} = -\frac{\partial u}{\partial x} + f(x,t), \quad x \in [-1,1],\ t > 0 \tag{1}$$

with periodic boundary conditions $u(-1,t) = u(1,t)$, and initial condition

$$u(x,0) = \begin{cases} 1 & x \in [-\frac{1}{2}, \frac{1}{2}] \\ 0 & \text{otherwise} \end{cases}$$

The function f is to be chosen later. In this project you are supposed to compare several discretizations for equation (1). We define the following:

1. Forward Difference:
$$D_+ u_k = \frac{u_{k+1} - u_k}{\Delta x}$$
2. Backward Difference:
$$D_- u_k = \frac{u_k - u_{k-1}}{\Delta x}$$
and
3. Centered Difference:
$$D_0 u_k = \frac{u_{k+1} - u_{k-1}}{2\Delta x}$$

Consider the following schemes:

1. Forward Time, Centered Space:
$$\frac{u_k^{n+1} - u_k^n}{\Delta t} = -D_0 u_k^n + f(x_k, t_n) \tag{2}$$
2. Leapfrog:
$$\frac{u_k^{n+1} - u_k^{n-1}}{2\Delta t} = -D_0 u_k^n + f(x_k, t_n) \tag{3}$$
3. Lax-Friedrichs:
$$\frac{u_k^{n+1} - \frac{1}{2}(u_{k+1}^n + u_{k-1}^n)}{\Delta t} = -D_0 u_k^n + f(x_k, t_n) \tag{4}$$

1. Find the accuracy of the three methods in space and time.
2. Choose an appropriate function $f(x,t)$ so that $u(x,t) = \cos(t)\sin(\pi x)$ is the actual solution.
3. Using the solution computed in the previous part, show that you recover the result from the previous question. What happens for the Forward Time, Centered Space?
4. Explain the behavior of the Forward Time, Centered Space. (*Hint: Von Neumann Analysis*).
5. Now set $f = 0$, and consider the original initial conditions. Solve the equation up to time $T = 1$. Plot the solution at time intervals $\Delta T = 0.1$, so that you can get an idea of the actual evolution. Solve the equation for several values of Δx and Δt to make sure your results don't change anymore when changing the grid size, or the time step. Plot the results for the different values to illustrate the convergence of the method.

Solution.

1. Consider the Forward Time, Centered Space (FTCS) scheme first,

$$\frac{u_k^{n+1} - u_k^n}{\Delta t} = -D_0 u_k^n + f(x_k, t_n).$$

We are supposed to compute the local truncation error as follows, to do this, we need to write the difference scheme in a form that directly models the derivatives and then substitute the exact solution in it. Rewrite the difference scheme as

$$\frac{u_k^{n+1} - u_k^n}{\Delta t} = -\frac{u_{k+1}^n - u_{k-1}^n}{2\Delta x} + f(x_k, t_n).$$

Substituting the exact solution in the above, the local truncation error at (x_k, t_n) is given as

$$\tau_k^n = \frac{u(x_k, t_{n+1}) - u(x_k, t_n)}{\Delta t} + \frac{u(x_{k+1}, t_n) - u(x_{k-1}, t_n)}{2\Delta x} - f(x_k, t_n). \quad (5)$$

Using Taylor series expansions around (x_k, t_n) we have

$$u(x_k, t_{n+1}) = u(x_k, t_n) + \Delta t\, u_t(x_k, t_n) + \frac{(\Delta t)^2}{2} u_{tt}(x_k, t_n) + O((\Delta t)^3) \quad (6)$$

$$u(x_{k+1},t_n) = u(x_k,t_n) + \Delta x\, u_x(x_k,t_n) + \frac{(\Delta x)^2}{2} u_{xx}(x_k,t_n) + O((\Delta x)^3) \tag{7}$$

$$u(x_{k-1},t_n) = u(x_k,t_n) - \Delta x\, u_x(x_k,t_n) + \frac{(\Delta x)^2}{2} u_{xx}(x_k,t_n) + O((\Delta x)^3) \tag{8}$$

Substituting (6), (7) and (8) in (5) we get

$$\begin{aligned}\tau_k^n &= u_t(x_k,t_n) + \frac{1}{2}u_{tt}(x_k,t_n)\Delta t + O((\Delta t)^2)\\ &\quad + u_x(x_k,t_n) + O((\Delta x)^2) - f(x_k,t_n).\end{aligned}$$

Since u is the exact solution, we have

$$\frac{\partial u}{\partial t} = -\frac{\partial u}{\partial x} + f(x,t).$$

So, we have

$$\tau_k^n = \frac{1}{2}u_{tt}(x_k,t_n)\Delta t + O((\Delta t)^2) + O((\Delta x)^2) = O(\Delta t + (\Delta x)^2).$$

Thus, the FTCS scheme is consistent and it is first order accurate in time and second order accurate in space.

We consider the Leapfrog scheme as above next, as a matter of fact, we consider only the derivative of time, since the order in space is second as above,

$$\frac{u_k^{n+1} - u_k^{n-1}}{2\Delta t} = -D_0 u_k^n + f(x_k,t_n)$$

i.e.,

$$\frac{u_k^{n+1} - u_k^{n-1}}{2\Delta t} = -\frac{u_{k+1}^n - u_{k-1}^n}{2\Delta x} + f(x_k,t_n)$$

Substituting the exact solution in the above, the local truncation error at (x_k,t_n) is given as

$$\tau_k^n = \frac{u(x_k,t_{n+1}) - u(x_k,t_{n-1})}{2\Delta t} + \frac{u(x_{k+1},t_n) - u(x_{k-1},t_n)}{2\Delta x} - f(x_k,t_n). \tag{9}$$

Using Taylor series expansions around (x_k, t_n) and Substituting the terms, we also observe that

$$\frac{\partial u}{\partial t} = -\frac{\partial u}{\partial x} + f(x,t).$$

Thus,

$$\tau_k^n = O((\Delta t)^2 + (\Delta x)^2).$$

We consider the Lax-Friedrichs scheme as above finally,

$$\frac{u_k^{n+1} - \frac{1}{2}(u_{k+1}^n + u_{k-1}^n)}{\Delta t} = -D_0 u_k^n + f(x_k, t_n)$$

i.e.,

$$\frac{u_k^{n+1} - \frac{1}{2}(u_{k+1}^n + u_{k-1}^n)}{\Delta t} = -\frac{u_{k+1}^n - u_{k-1}^n}{2\Delta x} + f(x_k, t_n).$$

The local truncation error at (x_k, t_n) is given as

$$\tau_k^n = \frac{u(x_k, t_{n+1}) - \frac{1}{2}(u(x_{k+1}, t_n) + u(x_{k-1}, t_n))}{\Delta t} + \frac{u(x_{k+1}, t_n) - u(x_{k-1}, t_n)}{2\Delta x} - f(x_k, t_n). \tag{10}$$

Using Taylor series expansions around (x_k, t_n) we have

$$u(x_k, t_{n+1}) = u(x_k, t_n) + \Delta t\, u_t(x_k, t_n) + \frac{(\Delta t)^2}{2} u_{tt}(x_k, t_n) + O((\Delta t)^3)$$

$$u(x_{k+1}, t_n) = u(x_k, t_n) + \Delta x\, u_x(x_k, t_n) + \frac{(\Delta x)^2}{2} u_{xx}(x_k, t_n) + O((\Delta x)^3)$$

$$u(x_{k-1}, t_n) = u(x_k, t_n) - \Delta x\, u_x(x_k, t_n) + \frac{(\Delta x)^2}{2} u_{xx}(x_k, t_n) + O((\Delta x)^3)$$

Substituting these extensions in (10) and noting that

$$\frac{\partial u}{\partial t} = -\frac{\partial u}{\partial x} + f(x,t)$$

and then, we get

$$\tau_k^n = \frac{\Delta t}{2} u_{tt}(x_k, t_n) - \frac{(\Delta x)^2}{2(\Delta t)} u_{xx}(x_k, t_n)$$
$$+ O(\frac{(\Delta x)^2}{\Delta t}) + O((\Delta t)^2) + O((\Delta x)^2).$$

Also letting the Courant number $r = \frac{\Delta t}{\Delta x}$ be constant, then $\frac{\Delta x}{\Delta t} = \frac{1}{r}$ and we have

$$\begin{aligned}\tau_k^n &= \frac{\Delta t}{2} u_{tt}(x_k, t_n) - \frac{\Delta x}{2r} u_{xx}(x_k, t_n) \\ &+ O(\frac{(\Delta x)^2}{r}) + O((\Delta x)^2) + O((\Delta t)^2) \\ &= O(\Delta t + \Delta x).\end{aligned}$$

In conclusion, the accuracy is as follow

Scheme	$\tau(\Delta x, \Delta t)$
FTCS	$O(\Delta t + (\Delta x)^2)$
Leapfrog	$O((\Delta t)^2 + (\Delta x)^2)$
Lax-Friedrichs	$O(\Delta t + \Delta x)$

2. Consider $u(x,t) = \cos(t)\sin(\pi x)$ solves the transport equation:

$$\frac{\partial u}{\partial t} = -\frac{\partial u}{\partial x} + f(x,t), \quad x \in [-1,1],\ t > 0$$

with periodic boundary conditions $u(-1,t) = u(1,t) = 0$, and initial condition

$$u(x,0) = \begin{cases} 1 & x \in [-\frac{1}{2}, \frac{1}{2}] \\ 0 & \text{otherwise} \end{cases}$$

Then,

$$\begin{aligned} f(x,t) &= \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} \\ &= \pi\cos(t)\cos(\pi x) - \sin(t)\sin(\pi x). \end{aligned}$$

We note that $f(x,t)$ satisfy with $f(-1,t) = f(1,t)$.

3. Using the solution computed in the previous part, show that you recover the result from the previous question. What happens for the Forward Time, Centered Space?

The result for exact solution is as follows,

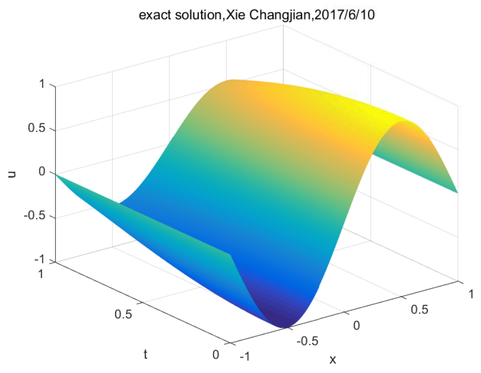

Figure 1: Do a plot of the exact solution up to time $T = 1$.

We deserve the numerical solution of three scheme, and we compare the error between them. We consider firstly the Lax-Friedrichs scheme, the matlab code is as follows,

```
%
% project 5
% LaxFriedrichs scheme
% Question 3 find the order
clear,clc
intervals=10^3;r=0.5;
% parameters
m=intervals-1;
h = 2/(m+1);
k = r*h;
T = 1;
% spatial mesh
x =-1:h:1;
% temporal mesh
nsteps = floor(T/k);
t=(0:k:nsteps*k)';
% initial condition
u=zeros(nsteps+1,m+2);
u(1,:)=sin(pi*x);
% f(x,t)=pi*cos(t)*cos(pi*x)-sin(t)*sin(pi*x)
f(1:nsteps+1,1:m+2)=pi*(cos(t))*cos(pi*x)...
```

```
    -sin(t)*sin(pi*x);
% periodic BC:
u(2,1)=1/2*(1-r)*u(1,2)+1/2*(1+r)*u(1,m+1)+k*f(1,1);
u(2,m+2)=1/2*(1-r)*u(1,2)+1/2*(1+r)*u(1,m+1)+k*f(1,m+2);
% interior x-points
u(2,2:m+1)=1/2*(1-r)*u(1,3:m+2)+1/2*(1+r)*u(1,1:m)+k*f(1,2:m+1);
u(3,1)=1/2*(1-r)*u(2,2)+1/2*(1+r)*u(2,m+1)+k*f(2,1);
u(3,m+2)=1/2*(1-r)*u(2,2)+1/2*(1+r)*u(2,m+1)+k*f(2,m+2);
u(3,2:m+1)=1/2*(1-r)*u(2,3:m+2)+1/2*(1+r)*u(2,1:m)+k*f(2,2:m+1);
for n=4:nsteps+1
u(n,1)=1/2*(1-r)*u(n-1,2)...
    +1/2*(1+r)*u(n-1,m+1)+k*f(n-1,1);
u(n,m+2)=1/2*(1-r)*u(n-1,2)...
    +1/2*(1+r)*u(n-1,m+1)+k*f(n-1,m+2);
u(n,2:m+1)=1/2*(1-r)*u(n-1,3:m+2)...
    +1/2*(1+r)*u(n-1,1:m)+k*f(n-1,2:m+1);
end
%plot
figure(1)
[X,T]=meshgrid(x,t);
mesh(X,T,u),hold on,
xlabel('x'),ylabel('t'),zlabel('u')
title('numerical solution for LaxFriedrichs,Xie Changjian,2017/6/10')
% exact solution
uk=zeros(nsteps+1,m+2);
uk(1:nsteps+1,1:m+2)=cos(t)*sin(pi*x);
%
figure(2)
[X,T]=meshgrid(x,t);
mesh(X,T,uk),hold on,
xlabel('x'),ylabel('t'),zlabel('u')
title('exact solution,Xie Changjian,2017/6/10')
%
figure(3)
plot(x,u(nsteps+1,:),'k--'),hold on,grid on
plot(x,uk(nsteps+1,:),'k.-'),hold on
legend('num','exact'),hold on
xlabel('x'),ylabel('u'),
title('the last time step'),
% find the error
format long
err=norm(u(nsteps+1,:)-uk(nsteps+1,:),inf)
%
figure(4)
% the last time step solution, find the order of err
dt=[10^(-3) 2*10^(-3) 4*10^(-3) 8*10^(-3) 10*10^(-3)];
dx=[2*10^(-3) 4*10^(-3) 8*10^(-3) 16*10^(-3) 20*10^(-3)];
err=[0.010565993093511 0.020958237147591 0.041232508570666 0.079814562078617
loglog(dt,err,'k*-'),hold on,grid on
xlabel('dt'),ylabel('err'),hold on
title('the plot of dt and err')
figure(5)
loglog(dx,err,'k-s'),hold on,grid on
xlabel('dx'),ylabel('err'),hold on
title('the plot of dx and err')
% find the slope_dt
r=sum((log(dt)).^2);
s=sum(log(dt));
t=sum(log(dt).*(log(err)));
p=sum(log(err));
slope_dt=(p*s-5*t)/(s^2-5*r);
% solve Cx=d
C=[sum((log(dt)).^2),sum(log(dt));sum(log(dt)),5];
d=[sum((log(dt)).*(log(err)));sum(log(err))];
xxx=C\d;
slope_dt=xxx(1);
intercept_dt=xxx(2);
slope_dt=vpa(slope_dt),intercept_dt=vpa(intercept_dt)
% find the slope_dx
r=sum((log(dx)).^2);
s=sum(log(dx));
t=sum(log(dx).*(log(err)));
p=sum(log(err));
slope_dx=(p*s-5*t)/(s^2-5*r);
% solve Cx=d
C=[sum((log(dx)).^2),sum(log(dx));sum(log(dx)),5];
```

```
d=[sum((log(dx)).*(log(err)));sum(log(err))];
xx=C\d;
slope_dx=xx(1);
intercept_dx=xx(2);
slope_dx=vpa(slope_dx),intercept_dx=vpa(intercept_dx)
```

As for Lax-Friedrichs, the result for numerical solution is as follows,

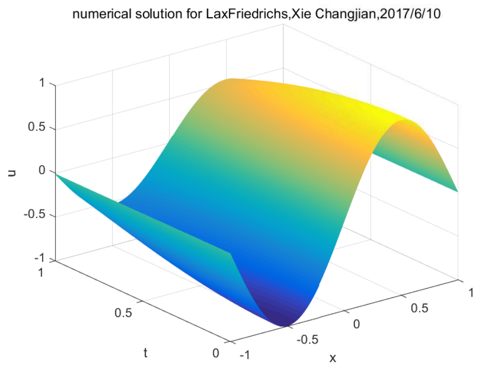

Figure 2: Do a plot of the numerical solution for Lax-Friedrichs scheme up to time $T = 1$.

The table of the error as the grid size Δx and the time step Δt is as follow,

the grid size Δx	the time step Δt	the mesh ratio r	err
2×10^{-3}	10^{-3}	0.5	0.010565993093511
4×10^{-3}	2×10^{-3}	0.5	0.020958237147591
8×10^{-3}	4×10^{-3}	0.5	0.041232508570666
16×10^{-3}	8×10^{-3}	0.5	0.079814562078617
20×10^{-3}	10×10^{-3}	0.5	0.098168334690265

Then, we can get the log-log plot of Δx and error as follow,

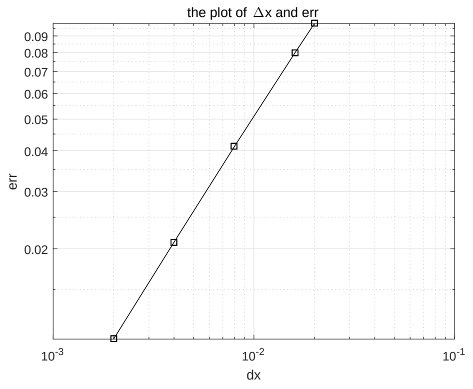

Figure 3: The order of error w.r.t Δx for Lax-Friedrichs scheme.

We can also get the log-log plot of Δt and error as follow,

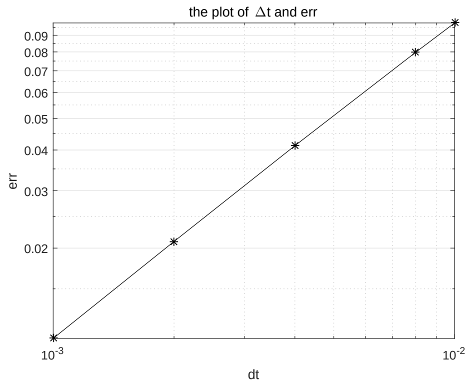

Figure 4: The order of error w.r.t Δt for Lax-Friedrichs scheme.

We compute the order of error as follows,

slope Δt	intercept Δt	slope Δx	intercept Δx
0.96823425	2.146484834	0.96823425144	1.4753559925

Then, we deserve the Lax-Friedrichs scheme is first order accurate in time and space.

In the following, we consider the Leapfrog scheme, since the matlab code is similar as above, we omit the detail, and we give the result of numerical solution for Leapfrog as follows,

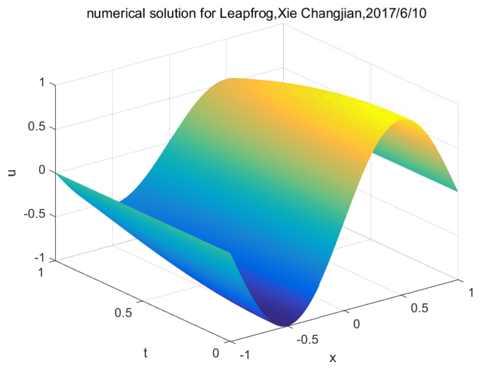

Figure 5: Do a plot of the numerical solution for Leapfrog scheme up to time $T = 1$.

As for Leapfrog scheme, the table of the error as the grid size Δx and the time step Δt is as follow,

the grid size Δx	the time step Δt	the mesh ratio r	err
2×10^{-3}	10^{-3}	0.5	$1.140976606284205 \times 10^{-5}$
4×10^{-3}	2×10^{-3}	0.5	$4.563919983568976 \times 10^{-5}$
8×10^{-3}	4×10^{-3}	0.5	$1.825673885328394 \times 10^{-4}$
16×10^{-3}	8×10^{-3}	0.5	$7.009488164758659 \times 10^{-4}$
20×10^{-3}	10×10^{-3}	0.5	0.001141284591331

Then, we can get the log-log plot of Δx or Δt and error as follow,

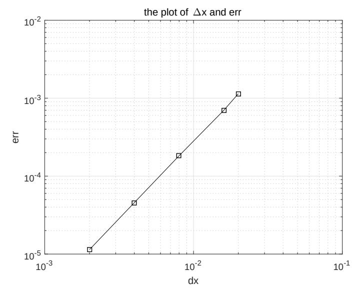

Figure 6: The log-log plot of Δx and error

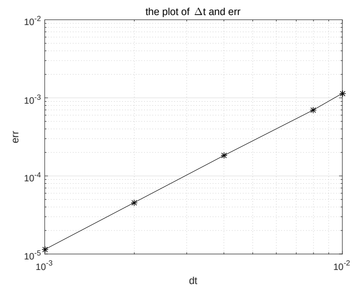

Figure 7: The log-log plot of Δt and error

We compute the order of error as follows,

slope Δt	intercept Δt	slope Δx	intercept Δx
1.99131846	2.37760200396	1.9913184609	0.99732522719

Then, we deserve the Leapfrog scheme is second order accurate in time and space. We note that the ratio of mesh $r \leq 1$ for the first two schemes, i.e., when $r \leq 1$, the Lax-Friedrichs and Leapfrog scheme are stable and convergent.

In the following, we consider the FTCS scheme, since the matlab code is similar as above, we also omit the detail, and if we set the relationship $\Delta t < (\Delta x)^2$, and the ratio of mesh is enough small, then, we give the result of numerical solution for FTCS as follows,

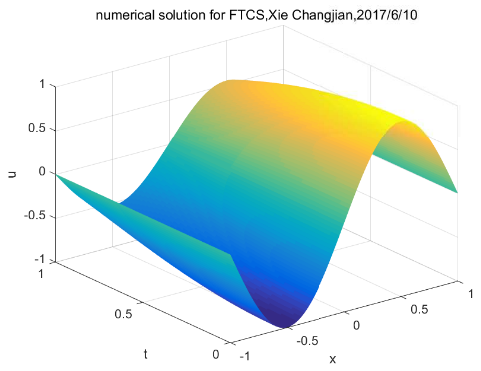

Figure 8: Do a plot of the numerical solution for FTCS scheme up to time $T = 1$, here we take $\Delta x = \frac{2}{160}$ and $\Delta t = \frac{1}{8000}$.

As for FTCS scheme, if$\Delta t < (\Delta x)^2$, the table of the error as the grid size Δx and the time step Δt is as follow,

the grid size Δx	the time step Δt	the mesh ratio r	err
$\frac{2}{6}$	0.1	0.3	0.398868102631832
$\frac{2}{20}$	0.005	0.05	0.029113597636520
$\frac{2}{40}$	0.00125	0.025	0.007199212990779
$\frac{2}{80}$	$\frac{1}{3200}$	0.0125	0.001793286258749
$\frac{2}{160}$	$\frac{1}{8000}$	0.01	$4.486758421282655 \times 10^{-4}$

Then, we can get the log-log plot of Δx or Δt and error as follow,

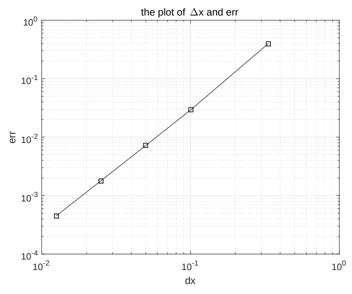

Figure 9: The log-log plot of Δx and error

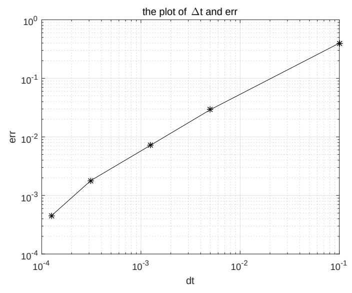

Figure 10: The log-log plot of Δt and error

We compute the order of error as follows,

slope Δt	intercept Δt	slope Δx	intercept Δx
0.99437936	1.54901452143	2.063645688059	1.286822705

Thus, the FTCS scheme is the first order accurate in time and the second order accurate in space.

If the relationship doesn't meet, i.e., $\Delta t > (\Delta x)^2$, the result is as follows,

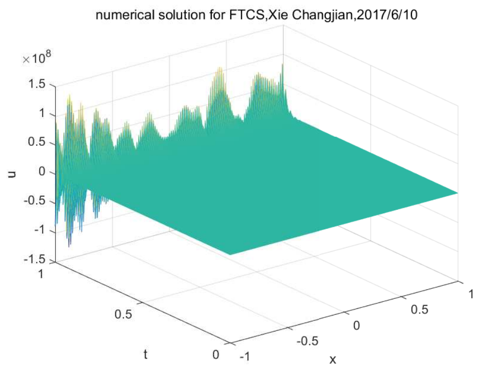

Figure 11: Do a plot of the numerical solution for FTCS scheme up to time $T = 1$, here we take $\Delta x = 0.004$ and $\Delta t = 0.002$.

We deserve the plot of the last time step as follows,

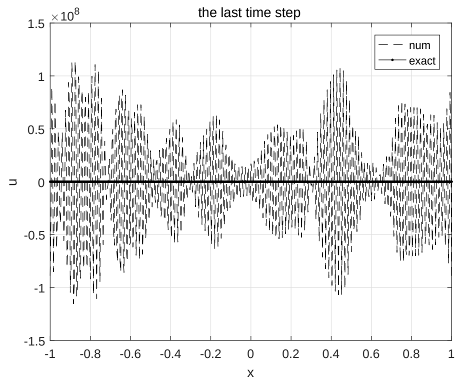

Figure 12: Do a plot of the numerical solution for FTCS scheme at the last time $t = T$, here we take $\Delta x = 0.004$ and $\Delta t = 0.002$.

Consider the FTCS as follows,

$$\frac{u_k^{n+1} - u_k^n}{\Delta t} = -\frac{u_{k+1}^n - u_{k-1}^n}{2\Delta x} + f(x_k, t_n). \tag{11}$$

$$\begin{aligned} \Rightarrow \quad u_k^{n+1} &= -\frac{\Delta t}{2\Delta x}(u_{k+1}^n - u_{k-1}^n) + u_k^n + \Delta t\, f(x_k, t_n) \\ &= -\frac{r}{2}(u_{k+1}^n - u_{k-1}^n) + u_k^n + \Delta t\, f(x_k, t_n). \end{aligned}$$

We will also consider weaker norms such as $\|\cdot\|_{1,h}$ and $\|\cdot\|_{2,h}$, where

$$\begin{aligned}\|u\|_{1,h} &= h\sum_k |u_k| \\ \|u\|_{2,h} &= (h\sum_k |u_k|^2)^{\frac{1}{2}}.\end{aligned}$$

we take multiply the u_k^n on both side of (11), then we get

$$2u_k^n(u_k^{n+1}-u_k^n)+r(u_{k+1}^n-u_{k-1}^n)u_k^n = 2\Delta t\, f(x_k,t_n)u_k^n.$$

$$\Rightarrow \quad (u_k^{n+1})^2 = (u_k^n)^2+(u_k^{n+1}-u_k^n)^2-r(u_{k+1}^n-u_{k-1}^n)u_k^n+2\Delta t\, f(x_k,t_n)u_k^n. \tag{12}$$

$$\begin{aligned}\Rightarrow \quad \|u^{n+1}\|_{2,h}^2 &= \|u^n\|_{2,h}^2 + h\sum_k (u_k^{n+1}-u_k^n)^2 \\ &\quad - rh\sum_k (u_{k+1}^n-u_{k-1}^n)u_k^n + 2h\Delta t\sum_k f(x_k,t_n)u_k^n \\ &= \|u^n\|_{2,h}^2 + h\sum_k (u_k^{n+1}-u_k^n)^2 + 2h\Delta t\sum_k f(x_k,t_n)u_k^n \\ &\geq \|u^n\|_{2,h}^2.\end{aligned}$$

Since the boundary condition exists, it follows that

$$\sum_k (u_{k+1}^n-u_{k-1}^n)u_k^n = u_{m+1}^n u_m^n - u_1^n u_0^n = 0.$$

We note that

$$f(x,t)u(x,t) = \frac{\pi}{2}\cos^2(t)\sin(2\pi x) - \frac{1}{2}\sin(2t)\sin^2(\pi x)$$

and

$$\sum_k \sin(2\pi x_k) = 0.$$

then

$$\sum_k f(x_k,t_n)u_k^n \geq 0.$$

Thus, the FTCS scheme is not strongly stable, that is, the result is not convergent but blow up.

4. Using Von Neumann Analysis to explain the behavior of the Forward Time, Centered Space.

   Let us approximate the initial condition by the discrete Fourier series and apply any of the numerical schemes studied till now to this approximate initial condition. Then the solution at any time t^n can be written as

$$u_k^n = \sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} \alpha_j (\gamma_j)^n \, e^{ikjh},$$

   where $\gamma_j \in \mathcal{C}$ is called the amplification coefficient of the k-th frequency (or harmonic). If $|\gamma_j| > 1$, then the numerical solution is likely to blow up with time.

   As for FTCS scheme, Consider first time step,

$$\begin{aligned} u_k^1 &= u_k^0 - \frac{r}{2}(u_{k+1}^0 - u_{k-1}^0) + \Delta t \, f(x_k, t_n) \\ &= \sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} \alpha_j \, e^{ikjh}[1 + \frac{r}{2}(e^{-ijh} - e^{ijh})] + \Delta t \, f(x_k, t_n). \end{aligned}$$

   But

$$\gamma_j = 1 + \frac{r}{2}(e^{-ijh} - e^{ijh}) = 1 - ir\sin(jh),$$

$$\Rightarrow \quad |\gamma_j| = [1 + r^2 \sin^2(jh)]^{\frac{1}{2}} \geq 1.$$

   Hence, the FTCS scheme is not strongly stable.

   However if we write the FTCS scheme as follows, note that $f(x_k, t_n)$ is a constant and it is known. So, we may as well set $f = 0$.

$$u_k^{n+1} - u_k^n = -\frac{r}{2}(u_{k+1}^n - u_{k-1}^n). \tag{13}$$

   Substituting (13) into (12), we get

$$\Rightarrow \quad (u_k^{n+1})^2 = (u_k^n)^2 + \left(\frac{r}{2}\right)^2 (u_{k+1}^n - u_{k-1}^n)^2 - r(u_{k+1}^n - u_{k-1}^n)u_k^n. \tag{14}$$

Summing up over all the equations

$$\begin{aligned}
\|u^{n+1}\|_{2,h}^2 &= \|u^n\|_{2,h}^2 + h\left(\frac{r}{2}\right)^2 \sum_k (u_{k+1}^n - u_{k-1}^n)^2 \\
&\le \|u^n\|_{2,h}^2 + h\left(\frac{r}{2}\right)^2 2\sum_k [(u_{k+1}^n)^2 + (u_{k-1}^n)^2] \\
&= \|u^n\|_{2,h}^2 + r^2\|u^n\|_{2,h}^2 \quad \text{assume } r^2 < \Delta t \\
&\le (1+\Delta t)\|u^n\|_{2,h}^2 \quad \text{if } \Delta t < h^2.
\end{aligned}$$

This yields

$$\|u^n\|_{2,h}^2 \le (1+\Delta t)^n \|u^0\|_{2,h}^2 \le e^T \|u^0\|_{2,h}^2 \quad \forall n \quad \text{s.t. } n\Delta t \le T,$$

which shows stability of FTCS scheme since

$$\|u^n\|_{2,h} \le e^{\frac{T}{2}} \|u^0\|_{2,h}.$$

Von Neumann stability tells us that if there exist $\beta \ge 0$ and a positive integer m such that, for suitable choices of Δt and h, we have $|\gamma_k| \le (1+\beta\Delta t)^{\frac{1}{m}}$, then the scheme is stable with respect to $\|\cdot\|_{2,h}$ with a stability constant $C_T = e^{\frac{\beta T}{m}}$. In particular, if $\beta = 0$ (i.e., $|\gamma_k| \le 1$), then the scheme is strongly stable.

5. Consider $f = 0$, and the original initial conditions

$$u(x,0) = \begin{cases} 1 & x \in [-\frac{1}{2}, \frac{1}{2}] \\ 0 & \text{otherwise} \end{cases}$$

In the following, we solve the equation up to time $T = 1$. Plot the solution at time intervals $\Delta T = 0.1$, so that you can get an idea of the actual evolution. Solve the equation for several values of Δx and Δt to make sure your results don't change anymore when changing the grid size, or the time step. Plot the results for the different values to illustrate the convergence of the method.

The matlab code for Leapfrog is that:

```
%
% project 5
% Leapfrog scheme
% Question 5
clear,clc
intervals=10;r=0.5;
% parameters
m=intervals-1;
h = 2/(m+1);
k = r*h;
T = 1;
% spatial mesh
x = [-1:h:1];
% temporal mesh
nsteps = floor(T/k);
% initial condition
u=zeros(nsteps+1,m+2);
for i=1:m+2
if(x(i)>=-1/2)&&(x(i)<=1/2)
    u(1,i)=1;
else
    u(1,i)=0;
end
end
% periodic BC:
u(2,1)=u(1,1)-r*(u(1,2)-u(1,m+1));
u(2,m+2)=u(1,m+2)-r*(u(1,2)-u(1,m+1));
% interior x-points
u(2,2:m+1)=u(1,2:m+1)-r*(u(1,3:m+2)-u(1,1:m));
u(3,1)=u(1,1)-r*(u(2,2)-u(2,m+1));
u(3,m+2)=u(1,m+2)-r*(u(2,2)-u(2,m+1));
u(3,2:m+1)=u(1,2:m+1)-r*(u(2,3:m+2)-u(2,1:m));
for n=4:nsteps+1
u(n,1)=u(n-2,1)-r*(u(n-1,2)-u(n-1,m+1));
u(n,m+2)=u(n-2,m+2)-r*(u(n-1,2)-u(n-1,m+1));
u(n,2:m+1)=u(n-2,2:m+1)-r*(u(n-1,3:m+2)-u(n-1,1:m));
end
%plot
t=0:k:nsteps*k;
figure(1)
```

```
[X,T]=meshgrid(x,t);
surf(X,T,u),hold on,
xlabel('x'),ylabel('t'),zlabel('u')
title('numerical solution for Leapfrog,Xie Changjian,2017/6/10')
```

the result for $\Delta t = 0.1$ is as follows,

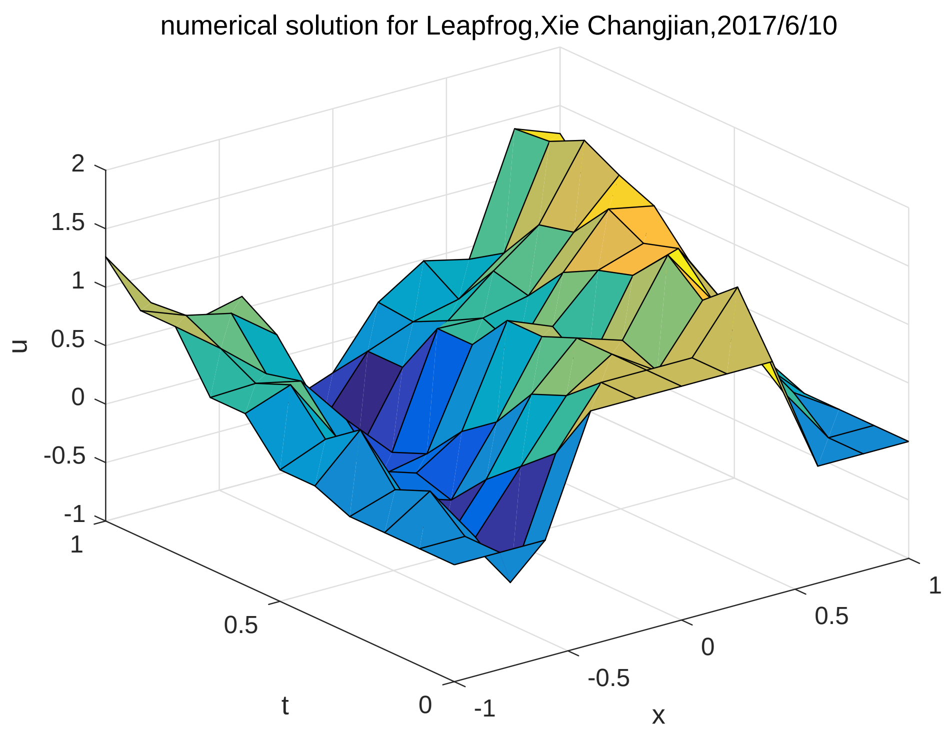

Figure 13: The solution at the time interval $\Delta t = 0.1$. Do a plot of the numerical solution up to time $T = 1$.

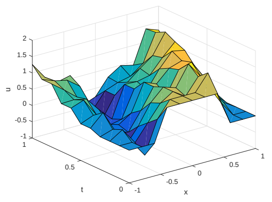

Figure 14: The grid size is $\Delta x = 0.2$, the time step $\Delta t = 0.1$.

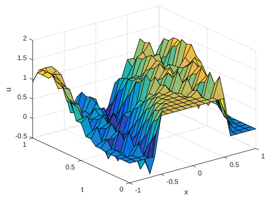

Figure 15: The grid size is $\Delta x = 0.1$, the time step $\Delta t = 0.05$.

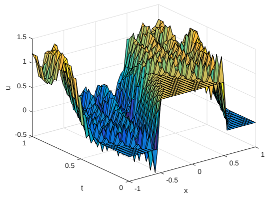

Figure 16: The grid size is $\Delta x = 0.05$, the time step $\Delta t = 0.025$.

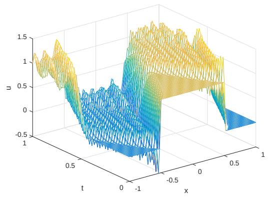

Figure 17: The grid size is $\Delta x = 0.025$, the time step $\Delta t = 0.0125$.

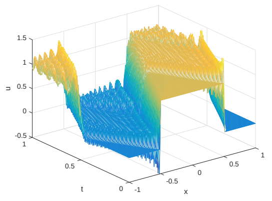

Figure 18: The grid size is $\Delta x = 0.0125$, the time step $\Delta t = 0.00625$.

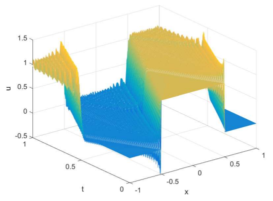

Figure 19: The grid size is $\Delta x = 0.00625$, the time step $\Delta t = 0.003125$.

As for the ratio of mesh $|r| \leq 1$, from these above six figures, we know the result don't change anymore when changing the grid size or the time step, if the stability condition doesn't meet, the result of Leapfrog isn't convergent, we can do the same Von Neumann analysis.

The Leapfrog numerical scheme is convergent.

The code of Lax-Friedrichs scheme is as follows,

```
%
% project 5
% LaxFriedrichs scheme
% Question 5
clear,clc
intervals=10;r=0.5;
% parameters
m=intervals-1;
h = 2/(m+1);
k = r*h;
T = 1;
% spatial mesh
x = [-1:h:1];
% temporal mesh
nsteps = floor(T/k);
% initial condition
u=zeros(nsteps+1,m+2);
for i=1:m+2
if(x(i)>=-1/2)&&(x(i)<=1/2)
    u(1,i)=1;
else
    u(1,i)=0;
end
end
% periodic BC:
u(2,1)=1/2*(1-r)*u(1,2)+1/2*(1+r)*u(1,m+1);
u(2,m+2)=1/2*(1-r)*u(1,2)+1/2*(1+r)*u(1,m+1);
% interior x-points
u(2,2:m+1)=1/2*(1-r)*u(1,3:m+2)+1/2*(1+r)*u(1,1:m);
u(3,1)=1/2*(1-r)*u(2,2)+1/2*(1+r)*u(2,m+1);
u(3,m+2)=1/2*(1-r)*u(2,2)+1/2*(1+r)*u(2,m+1);
u(3,2:m+1)=1/2*(1-r)*u(2,3:m+2)+1/2*(1+r)*u(2,1:m);
for n=4:nsteps+1
```

```
u(n,1)=1/2*(1-r)*u(n-1,2)+1/2*(1+r)*u(n-1,m+1);
u(n,m+2)=1/2*(1-r)*u(n-1,2)+1/2*(1+r)*u(n-1,m+1);
u(n,2:m+1)=1/2*(1-r)*u(n-1,3:m+2)+1/2*(1+r)*u(n-1,1:m);
end
%plot
t=0:k:nsteps*k;
figure(1)
[X,T]=meshgrid(x,t);
mesh(X,T,u),hold on,
xlabel('x'),ylabel('t'),zlabel('u')
title('numerical solution for LaxFriedrichs,Xie Changjian,2017/6/10')
```

The result for $\Delta t = 0.1$ is as follows,

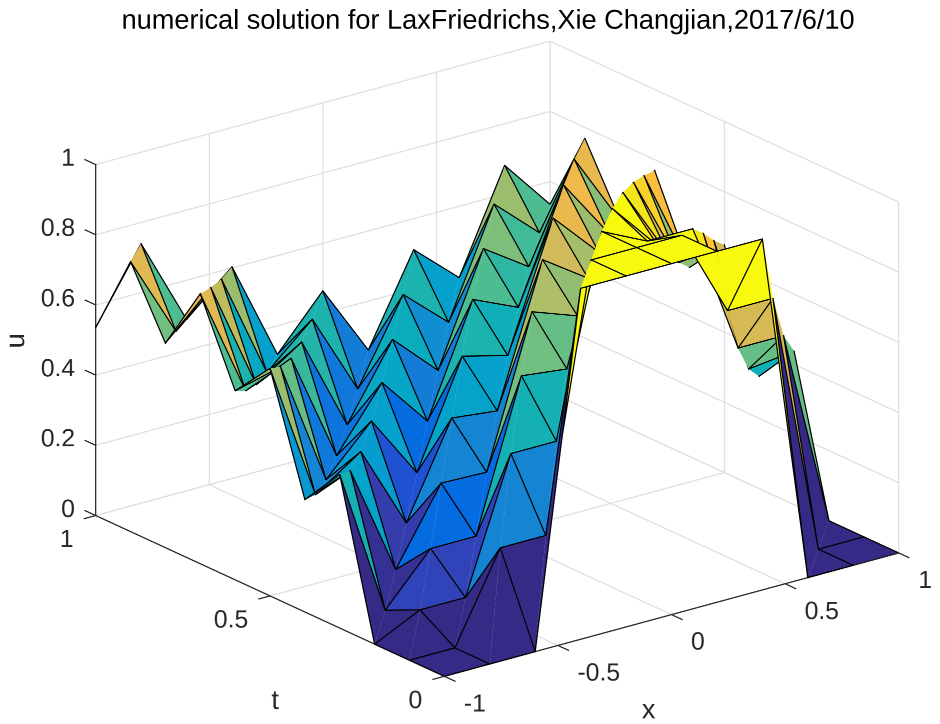

Figure 20: The solution at the time interval $\Delta t = 0.1$. Do a plot of the numerical solution up to time $T = 1$.

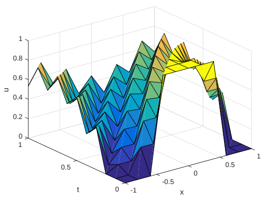

Figure 21: The grid size is $\Delta x = 0.2$, the time step $\Delta t = 0.1$.

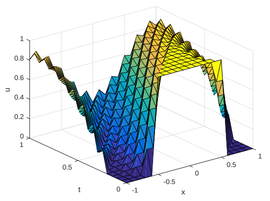

Figure 22: The grid size is $\Delta x = 0.1$, the time step $\Delta t = 0.05$.

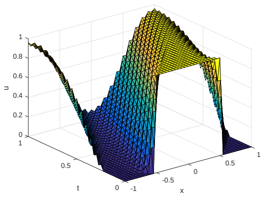

Figure 23: The grid size is $\Delta x = 0.05$, the time step $\Delta t = 0.025$.

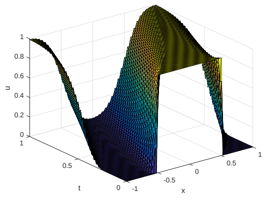

Figure 24: The grid size is $\Delta x = 0.025$, the time step $\Delta t = 0.0125$.

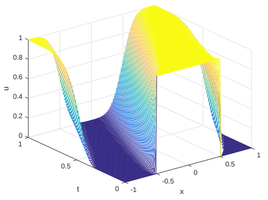

Figure 25: The grid size is $\Delta x = 0.0125$, the time step $\Delta t = 0.00625$.

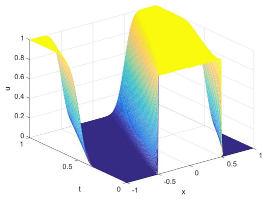

Figure 26: The grid size is $\Delta x = 0.00625$, the time step $\Delta t = 0.003125$.

As for the ratio of mesh $|r| \leq 1$, from these above six figures, we know the result don't change anymore when changing the grid size or the time step. If the stability condition doesn't meet, the result of Leapfrog isn't convergent, we can do the same Von Neumann analysis.

The Lax-Friedrichs numerical scheme is convergent when $|r| \leq 1$.

The code of FTCS scheme is as follows,

```
%
% project 5
% FTCS scheme
% Question 5
clear,clc
intervals=10;r=0.5;
% parameters
m=intervals-1;
h = 2/(m+1);
k = r*h;
T = 1;
% spatial mesh
x = [-1:h:1];
% temporal mesh
nsteps = floor(T/k);
% initial condition
u=zeros(nsteps+1,m+2);
for i=1:m+2
if(x(i)>=-1/2)&&(x(i)<=1/2)
    u(1,i)=1;
else
    u(1,i)=0;
end
end
% periodic BC:
u(2,1)=u(1,1)-r/2*(u(1,2)-u(1,m+1));
u(2,m+2)=u(1,m+2)-r/2*(u(1,2)-u(1,m+1));
% interior x-points
u(2,2:m+1)=u(1,2:m+1)-r/2*(u(1,3:m+2)-u(1,1:m));
u(3,1)=u(2,1)-r/2*(u(2,2)-u(2,m+1));
u(3,m+2)=u(2,m+2)-r/2*(u(2,2)-u(2,m+1));
u(3,2:m+1)=u(2,2:m+1)-r/2*(u(2,3:m+2)-u(2,1:m));
for n=4:nsteps+1
u(n,1)=u(n-1,1)-r/2*(u(n-1,2)-u(n-1,m+1));
u(n,m+2)=u(n-1,m+2)-r/2*(u(n-1,2)-u(n-1,m+1));
```

```
u(n,2:m+1)=u(n-1,2:m+1)-r/2*(u(n-1,3:m+2)-u(n-1,1:m));
end

%plot
t=0:k:nsteps*k;
figure(1)
[X,T]=meshgrid(x,t);
mesh(X,T,u),hold on,
xlabel('x'),ylabel('t'),zlabel('u')
title('numerical solution for FTCS,Xie Changjian,2017/6/10')
```

The result for $\Delta t = 0.1$ is as follows,

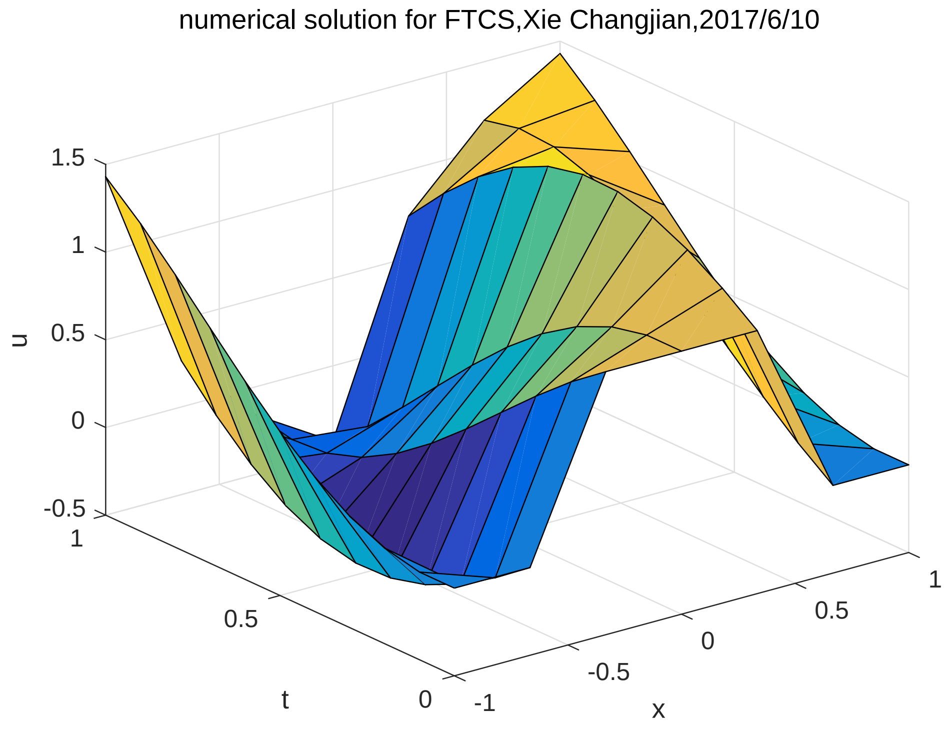

Figure 27: The solution at the time interval $\Delta t = 0.1$. Do a plot of the numerical solution up to time $T = 1$.

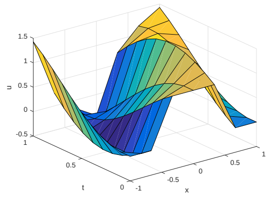

Figure 28: The grid size is $\Delta x = \frac{2}{6}$, the time step $\Delta t = 0.1$.

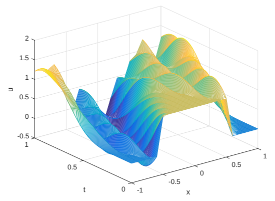

Figure 29: The grid size is $\Delta x = \frac{2}{20}$, the time step $\Delta t = 0.005$.

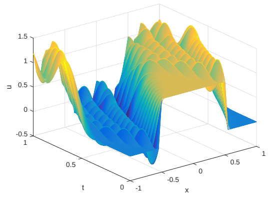

Figure 30: The grid size is $\Delta x = \frac{2}{40}$, the time step $\Delta t = 0.00125$.

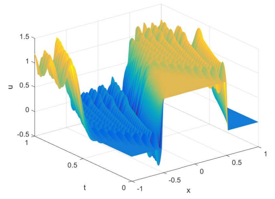

Figure 31: The grid size is $\Delta x = \frac{2}{80}$, the time step $\Delta t = \frac{1}{3200}$.

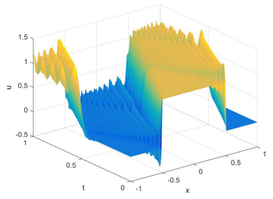

Figure 32: The grid size is $\Delta x = \frac{2}{160}$, the time step $\Delta t = \frac{1}{8000}$.

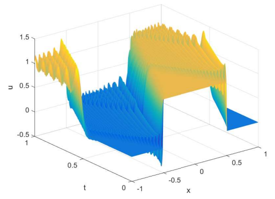

Figure 33: The grid size is $\Delta x = \frac{2}{200}$, the time step $\Delta t = \frac{1}{2\times 10^4}$.

As for $\Delta t < (\Delta x)^2$, and the ratio of mesh r is enough small, it requires the time step strictly, we know the result don't change anymore when changing the grid size or the time step. Then, the result is stable and convergent. But if the stability condition doesn't meet, the result of FTCS scheme is not convergent but blow up.