Matteo Polimeno

February 13, 2018

MATH693b
Homework 01

**Dr.Blomgren**

**Solution for exercise 1.3.1 from Strikveda**
The one way wave equation is given by

$$u_t + u_x = 0,$$

and we have $x \in [-1, 3]$ and $t \in [0, 2.4]$. Initial data:

$$u(0, x) = \begin{cases} \cos^2(\pi x) & \text{if } |x| < \frac{1}{2} \\ 0 & \text{otherwise,} \end{cases}$$

and the boundary data is $u(t, -1) = 0$.
We will use different schemes to approximate the solution of the given PDE.
Below is the function that we implemented to solve the one-way wave equation and whose solution we compare to the various schemes (Part a through d).

```
%one-way wave equation (hyperbolic PDE)
function J = hwe(x)

if abs(x) <= .5
    J = cos(pi*x)^2;
else
    J = 0;
end


end
```

# 1 Part a

First we use the Forward-time Backward-space scheme given by

$$\frac{v_m^{n+1} - v_m^n}{k} + a\frac{v_m^n - v_{m-1}^n}{h} = 0,$$

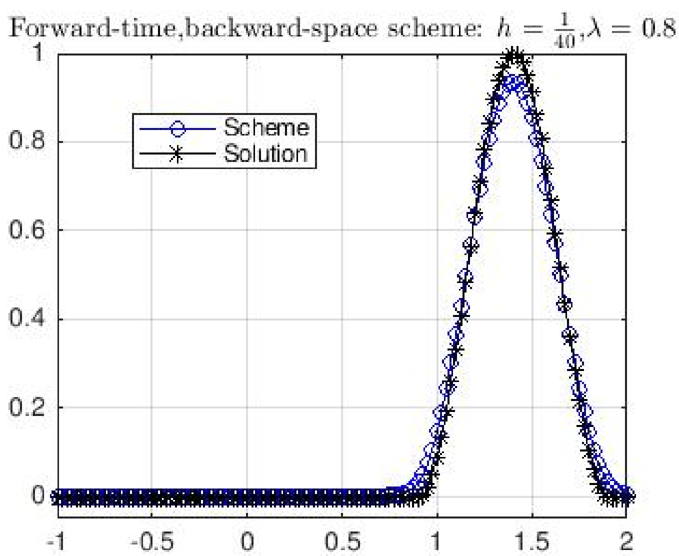
solving for $v_m^{n+1}$ and making $k/h = \lambda$ we get

$$v_m^{n+1} = a\lambda v_{m-1}^n - v_m^n(a\lambda - 1)$$

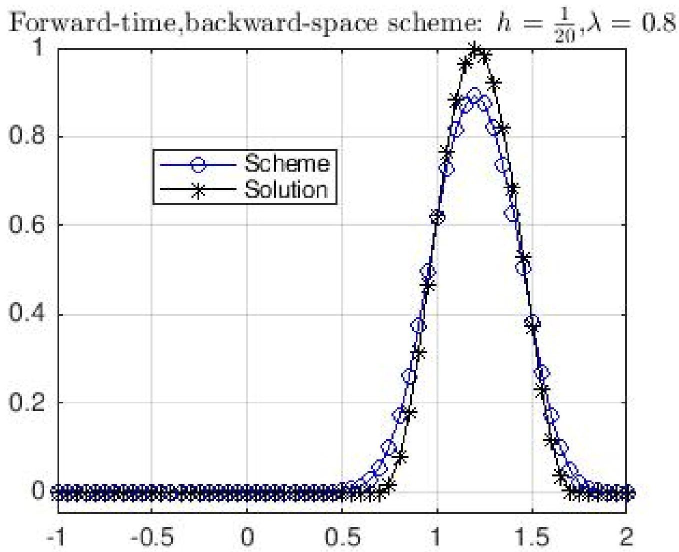
### Matlab Code for Part a

```
%Forward−time backward−space scheme
clear all
clc

lambda = .8;
h = 1/40;
xd = −1:h:3;
p = length(xd);
k = lambda*h;
td = 0:k:2.4;
q=(length(td));
ud = zeros(length(td),1);


[X,Y] = meshgrid(xd,td);

for i = 1:p
    if abs(xd(i)) <= .5
        u(1,i) = cos(pi*xd(i))^2;
    else
        u(1,i) = 0;
    end
end

for i = 1:q
    u(i,1) = 0;
end


for i = 1:q−1
    for j = 1:p−1
        u(i+1,j+1) = (1−lambda)*u(i,j+1) + lambda*u(i,j);
    end
end

```

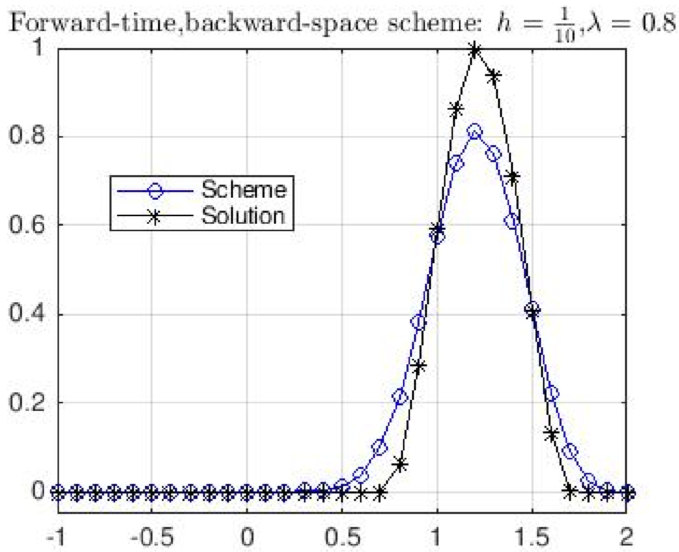
```
for i = 1:q
    v(i,1) = 0;
end


for i = 1:q-1
    for j = 1:p-1
        v(i+1,j+1) = hwe((xd(j) -  td(i)));
    end
end

for i = 1:q-50
    plot(xd,u(i,:),'b-o',xd,v(i,:),'k-*');
    ylim([-0.05,1])
    xlim([-1,2])
    grid on
   M(i) = getframe;
end

title('Forward-time backward-space scheme for $h=\frac{1}{40}$ and $\lambda=0.8$', 'Interpreter', 'latex')
```




Thus, we see that for $h = \frac{1}{40}$ and $\lambda = 0.8$ the scheme is a useful scheme as it approximates the solution pretty well. We lose something at the peak as the scheme shrinks and the there is some noise around $t = 1$, where the scheme overestimates the solution. Yet, overall, the result is pretty good.




Thus, we see that for $h = \frac{1}{20}$ and $\lambda = 0.8$ the scheme is a quite less useful. The overall behavior of the solution is still approximated, but the scheme shrinks a lot more and we lose more information at the peak.




Thus, we see that for $h = \frac{1}{10}$ and $\lambda = 0.8$ the scheme is a again less useful. The overall behavior of the solution is still approximated, but the scheme shrinks a lot more and we lose more information at the peak. Also the solution and the schemes have several cusps.

## 2 Part b

Now we use the Forward-time central-space scheme

$$\frac{v_m^{n+1} - v_m^n}{k} + a\frac{v_{m+1}^n - v_{m-1}^n}{h} = 0,$$

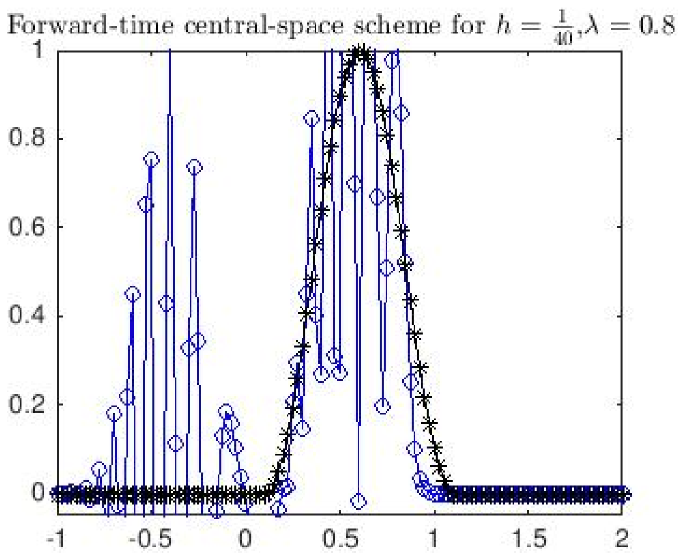
again we solve for $v_m^{n+1}$

$$v_m^{n+1} = v_m^n - \frac{1}{2}a\lambda v_{m+1}^n + \frac{1}{2}a\lambda v_{m-1}^n$$

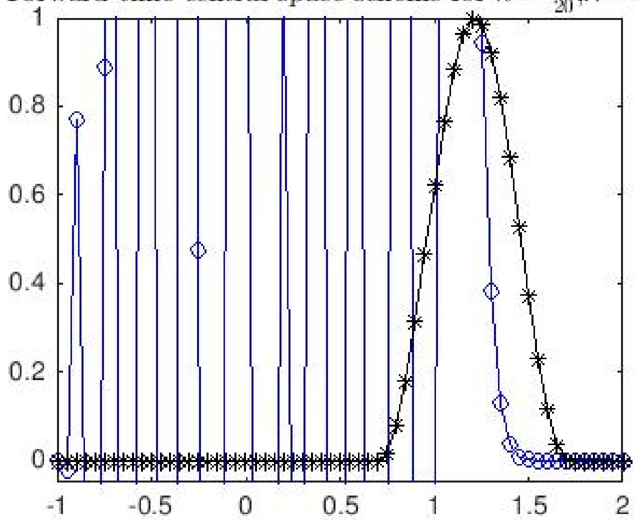
### Matlab Code for Part b

```
%Forward-time central-space scheme
clear all
clc

lambda = .8;
h = 1/40;
xd = -1:h:3;
p = length(xd);
k = lambda*h;
td = 0:k:2.4;
q=(length(td));
ud = zeros(length(td),1);


[X,Y] = meshgrid(xd,td);

for i = 1:p
    if abs(xd(i)) <= .5
        u(1,i) = cos(pi*xd(i))^2;
    else
        u(1,i) = 0;
    end
end

for i = 1:q
    u(i,1) = 0;
end


for i = 1:q-1
    for j = 1:p-2
        u(i+1,j+1) = -lambda*((u(i,j+2) - u(i,j)))/2 + u(i,j+1);
    end
end

```

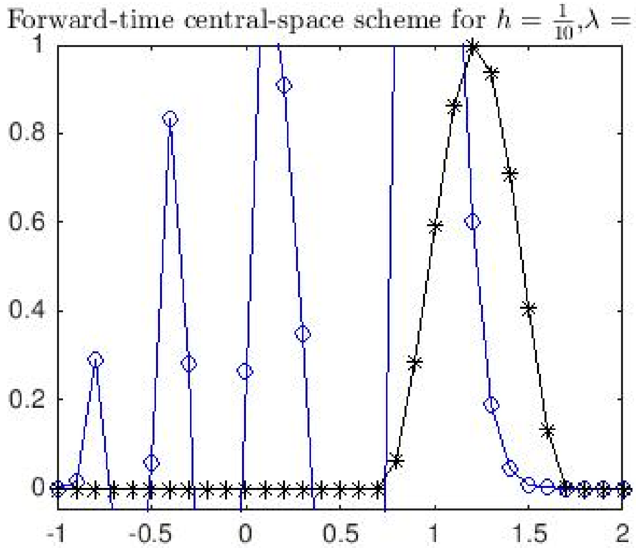
```
for i = 1:q-1
    for j = 1:p-1
        v(i+1,j+1) = hwe((xd(j) -  td(i)));
    end
end

u(:,p) = u(:,p-1);


for i = 1:q-90
    plot(xd,u(i,:),'b-o',xd,v(i,:),'k-*');
    ylim([-0.05,1])
    xlim([-1,2])
   M(i) = getframe;
end

title('Forward-time central-space scheme for $h=\frac{1}{40}$ and $\lambda=0.8$', 'Interpreter', 'latex')
```



Forward-time central-space scheme for $h = \frac{1}{20}, \lambda = 0.8$



Forward-time central-space scheme for $h = \frac{1}{10}, \lambda = 0.8$



Therefore we see that in this case, regardless of the value of $h$, the scheme is useless: it blows up almost right away. As we decrease $h$ the approximation gets worse and worse.

# 3 Part c

Now we use the Lax-Friedrichs scheme

$$\frac{v_m^{n+1} - \frac{1}{2}(v_{m+1}^n + v_{m-1}^n)}{k} + a\frac{v_{m+1}^n - v_{m-1}^n}{2h} = 0,$$

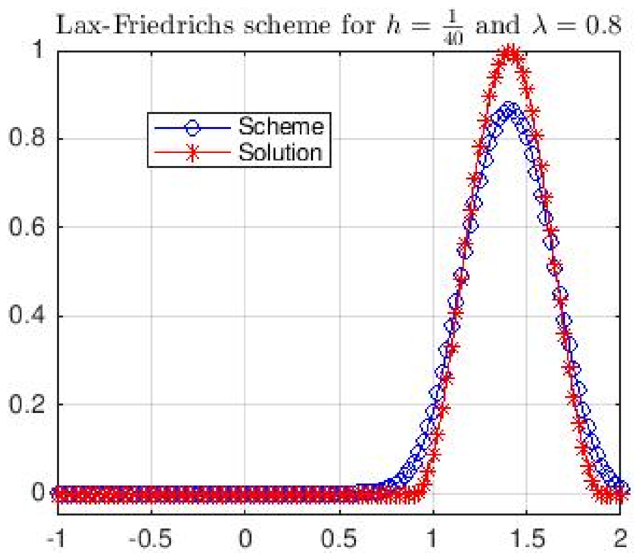
again we solve for $v_m^{n+1}$

$$v_m^{n+1} = v_{m+1}^n\left(\frac{1}{2} - a\lambda\right) + v_{m-1}^n\left(\frac{1}{2} + a\lambda\right).$$

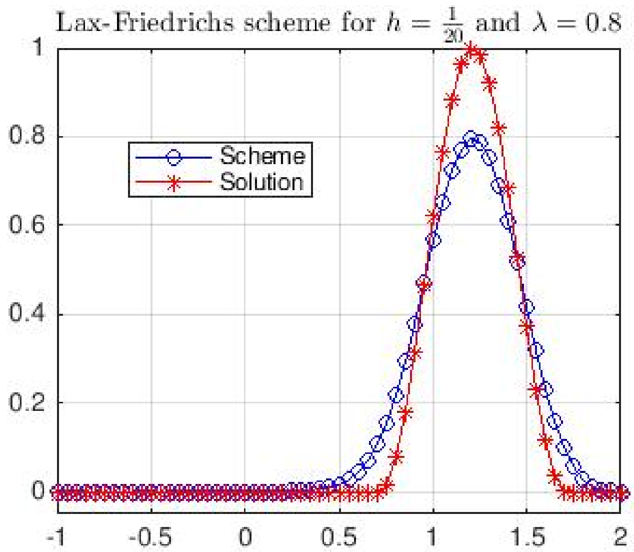
### Matlab Code for Part c

```
%Lax-Friedrichs scheme
clear all
clc

lambda = .8;
h = 1/40;
xd = -1:h:3;
p = length(xd);
k = lambda*h;
td = 0:k:2.4;
q=(length(td));
ud = zeros(length(td),1);

[X,Y] = meshgrid(xd,td);

for i = 1:p
    if abs(X(1,i)) <= .5
        u(1,i) = cos(pi*X(1,i))^2;
    else
        u(1,i) = 0;
    end
end

for i = 1:q
    u(i,1) = 0;
end


for i = 1:q-1
    for j = 1:p-2
        u(i+1,j+1) = -lambda*((u(i,j+2) - u(i,j))/2) + ((u(i,j+2) + u(i,j))/2);
    end
end

```

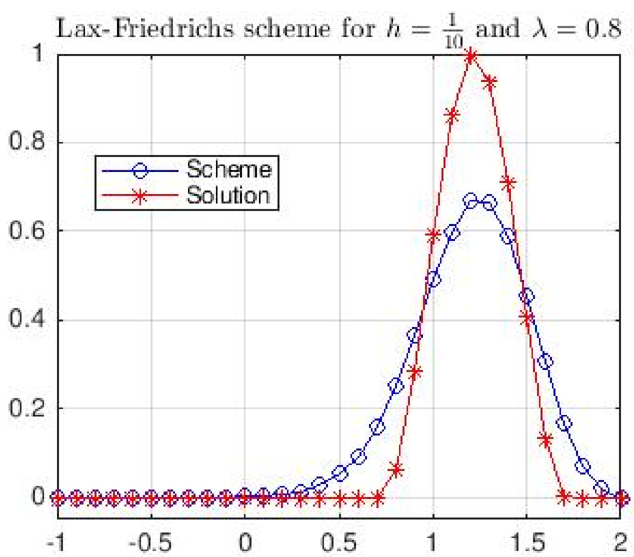
```
for i = 1:q-1
    for j = 1:p-1
        v(i+1,j+1) = hwe((xd(j) - td(i)));
    end
end

u(:,p) = u(:,p-1);

for i = 1:q-50
    plot(xd,u(i,:),'b-o',xd,v(i,:),'r-*');
    ylim([-0.05,1])
    xlim([-1,2])
    grid on
    M(i) = getframe;
end

title('Lax-Friedrichs scheme for $h=\frac{1}{40}$ and $\lambda=0.8$', '
    Interpreter','latex')
```



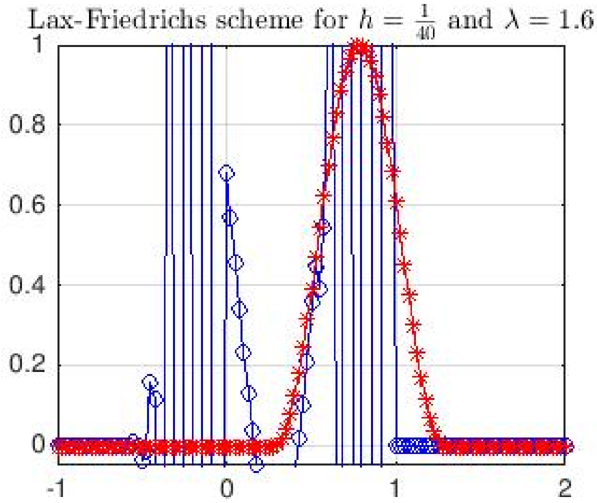

Thus, our scheme is a decent approximation for the general behavior of the solution, but we lose some information at the peak as the scheme shrinks. The scheme is useful, even though not great.

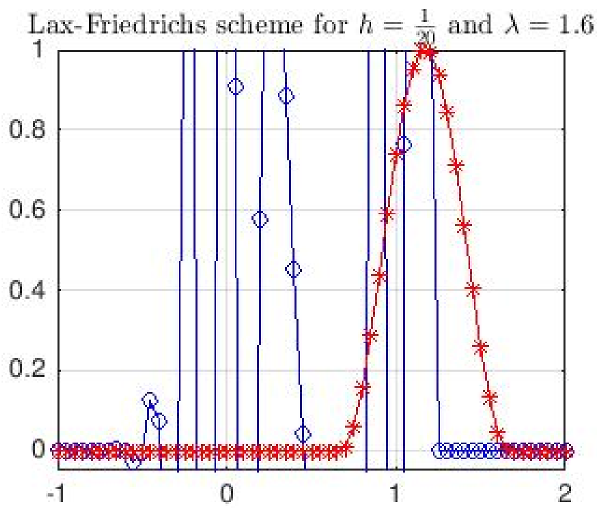


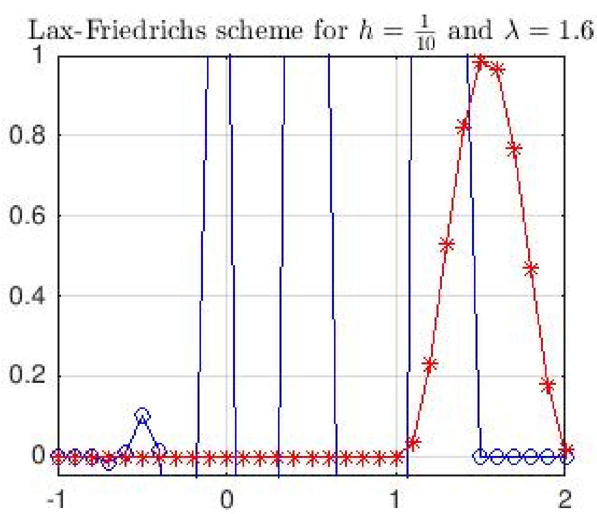

Now the scheme, even though it follows the solution in its behavior, is basically useless. We lose at lot of information at the peak and at the bottom around $t = 0.7$ and $t = 1.7$.




For $h = \frac{1}{10}$ the scheme is useless as it shrinks a lot and does not approximate the solution.


Lax-Friedrichs scheme for $h = \frac{1}{40}$ and $\lambda = 1.6$
1
0.8
0.6
0.4
0.2
0
-1
0
1
2


Lax-Friedrichs scheme for $h = \frac{1}{20}$ and $\lambda = 1.6$
1
0.8
0.6
0.4
0.2
0
-1
0
1
2


Lax-Friedrichs scheme for $h = \frac{1}{10}$ and $\lambda = 1.6$
1
0.8
0.6
0.4
0.2
0
-1
0
1
2

We see that doubling the value of $\lambda$ makes the situation worse and thus the scheme useless. There is no approximation of the solution and the scheme blows up right away. Changing the value of $h$ for $\lambda = 1.6$ makes little to no difference in terms of the usefulness of the scheme, but it provides some visuals with regards of how fast the scheme blows up.

## 4 Part d

Now we will use the Leap-Frog scheme

$$\frac{v_m^{n+1} - v_m^{n-1}}{2k} + a\frac{v_{m+1}^n - v_{m-1}^n}{2h} = 0.$$

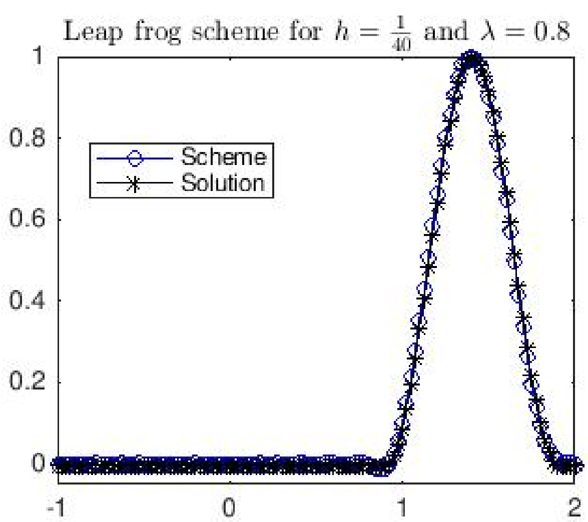
Solving for $v_m^{n+1}$ yields

$$v_m^{n+1} = v_m^{n-1} - a\lambda v_{m+1}^n + a\lambda v_{m-1}^n$$

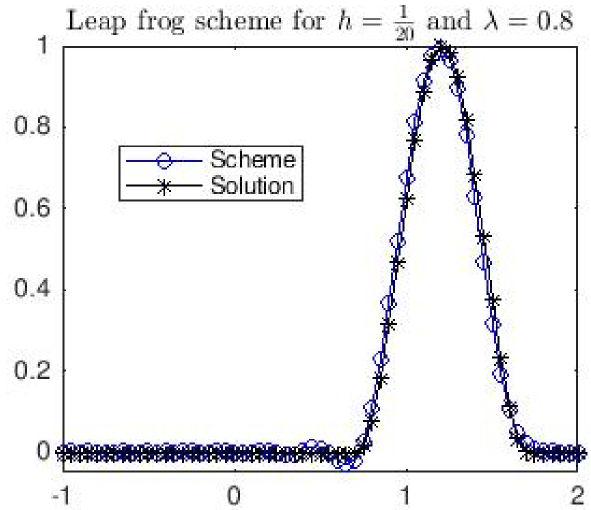
### 4.1 Matlab code for Part d

```
%Leapfrog scheme
clear all
clc

lambda = .8;
h = 1/40;
xd = -1:h:3;
p = length(xd);
k = lambda*h;
td = 0:k:2.4;
q=(length(td));
ud = zeros(length(td),1);


[X,Y] = meshgrid(xd,td);

for i = 1:p
    if abs(xd(i)) <= .5
        u(1,i) = cos(pi*xd(i))^2;
    else
        u(1,i) = 0;
    end
end

for i = 1:q
    u(i,1) = 0;
end

for i = 1:q
     for j = 1:p-1
         u(i+1,j+1) = (1-lambda)*u(i,j+1) + lambda*u(i,j);
     end
end


```

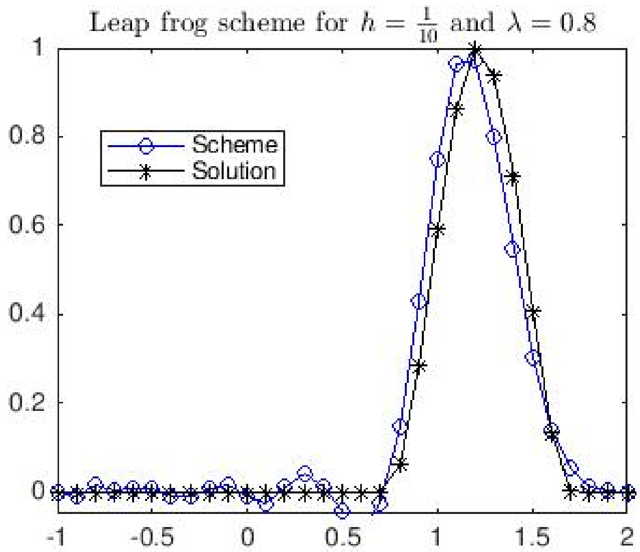
```matlab
for i = 1:q-1 %leapfrog
    for j = 1:p-2
        u(i+2,j+1) = -lambda*(u(i+1,j+2) - u(i+1,j)) + u(i,j+1);
    end
end

for i = 1:q-1
    for j = 1:p-1
         v(i+1,j+1) = hwe((xd(j) -  td(i)));
    end
end

u(:,p) = u(:,p-1);

for i = 1:q-50
    plot(xd,u(i,:),'b-o',xd,v(i,:),'k-*');
    ylim([-0.05,1])
    xlim([-1,2])
   M(i) = getframe;
end

title('Leap frog scheme for $h=\frac{1}{40}$ and $\lambda=0.8$', '
    Interpreter','latex')
```




Here we see that the scheme is extremely useful. It approximates the solution almost perfectly.




The scheme is still useful and the approximation is quite good, although there is a little noise at the bottom before the increasing of the solution.




In this case there is a lot of noise between the solution and the scheme and it is not useful.

**Solution for exercise 1.4.1 from Strikveda**
We are given

$$P\phi = \phi_t + a\phi_x,$$

thus, for the forward-time-central space scheme, the difference operator $P_{k,h}\phi$ we have

$$P_{k,h}\phi = \frac{\phi_m^{n+1} - \phi_m^n}{k} + a\frac{\phi_{m+1}^n - \phi_{m-1}^n}{2h}.$$

We Taylor-expand

$$\phi_m^{n+1} \approx \phi_m^n + k\phi_t + \frac{k^2}{2}\phi_{tt} + \mathcal{O}(k^3),$$

$$\phi_{m+1}^n \approx \phi_m^n + h\phi_x + \frac{h^2}{2}\phi_{xx} + \mathcal{O}(h^3),$$

$$\phi_{m-1}^n \approx \phi_m^n - h\phi_x + \frac{h^2}{2}\phi_{xx} + \mathcal{O}(h^3).$$

Now, we plug it into our difference operator

$$P_{k,h}\phi = \frac{\phi_m^n + k\phi_t + \frac{k^2}{2}\phi_{tt} + \mathcal{O}(k^3) - \phi_m^n}{k} + a\frac{\phi_m^n + h\phi_x + \frac{h^2}{2}\phi_{xx} + \mathcal{O}(h^3) - \phi_m^n - h\phi_x + \frac{h^2}{2}\phi_{xx} + \mathcal{O}(h^3)}{2h}$$

$$\Rightarrow P_{k,h}\phi = \phi_t + \frac{k}{2}\phi_{tt} + a\phi_x + \mathcal{O}(k^3) + \mathcal{O}(h^3).$$

Thus, since we know that a scheme is consistent if

$$P\phi - P_{k,h}\phi \longrightarrow 0 \; as \; k, h \longrightarrow 0,$$

we solve

$$P\phi - P_{k,h}\phi = \phi_t + a\phi_x - \phi_t - \frac{k}{2}\phi_{tt} - a\phi_x + \mathcal{O}(k^3) + \mathcal{O}(h^3)$$

which yields, after simplification,

$$P\phi - P_{k,h}\phi = -\frac{k}{2}\phi_{tt} + \mathcal{O}(k^3) + \mathcal{O}(h^3),$$

which goes to 0 as $k, h \to 0$. Therefore the scheme is consistent.

**Solution for exercise 1.5.1 from Strikveda**
We will prove that a scheme of the form

$$v_m^{n+1} = \alpha v_{m+1}^n + \beta v_{m-1}^n$$

is stable if $|\alpha| + |\beta| \leq 1$.
Thus, we have

$$\sum_{m=-\infty}^{\infty} |v_m^{n+1}|^2 = \sum_{m=-\infty}^{\infty} |\alpha v_{m+1}^n + \beta v_{m-1}^n|^2$$

$$\leq \sum_{m=-\infty}^{\infty} |\alpha|^2 |v_{m+1}^n|^2 + 2|\alpha||\beta||v_{m+1}^n||v_{m-1}| + |\beta|^2 + |v_{m-1}^2|^2$$

$$\leq \sum_{m=-\infty}^{\infty} |\alpha|^2 |v_{m+1}^n|^2 + |\alpha||\beta|(|v_{m+1}^n|^2 + |v_{m-1}^n|^2) + |\beta|^2 |v_{m-1}^n|^2$$

The last sum can be split over the terms $m+1$ and $m-1$ and we can also shift the index as the upper and lower limits of our sum will not be affected and make everything in terms of $m$. Thus

$$= \sum_{m=-\infty}^{\infty} |\alpha|^2 |v_m^n|^2 |\alpha||\beta||v_m^n|^2 + \sum_{m=-\infty}^{\infty} |\beta||v_m^n|^2 + |\alpha||\beta|v_m^n|^2$$

$$= \sum_{m=-\infty}^{\infty} |\alpha|^2 |v_m^n|^2 |\alpha||\beta||v_m^n|^2 + \sum_{m=-\infty}^{\infty} |\beta||v_m^n|^2 + |\alpha||\beta|v_m^n|^2$$

$$= \sum_{m=-\infty}^{\infty} (|\alpha|^2 + 2|\alpha||\beta| + |\beta|^2)|v_m^n|^2$$

$$= (|\alpha| + |\beta|)^2 \sum_{m=-\infty}^{\infty} |v_m^n|^2$$

So we have the relationship

$$|v_m^{n+1}|^2 \leq (|\alpha| + |\beta|)^2 \sum_{m=-\infty}^{\infty} |v_m^n|^2,$$

and, since this applies for all $n$, then we have that

$$|v_m^n|^2 \leq (|\alpha| + |\beta|)^{2n} \sum_{m=-\infty}^{\infty} |v_m^0|^2.$$

Therefore, if $|\alpha| + |\beta| \leq 1$, then the scheme will be stable.
For the Lax-Friedrichs scheme we have

$$\frac{v_m^{n+1} - \frac{1}{2}v_{m+1}^n - \frac{1}{2}v_{m-1}^n}{k} + a\frac{v_{m+1}^n - v_{m-1}^n}{2h} = 0$$

$$\Rightarrow 2hv_m^{n+1} - hv_{m+1}^n - hv_{m-1}^n + akv_{m+1}^n - akv_{m-1}^n = 0.$$

We solve for $v_m^{n+1}$ and, to simplify our notation, call $\frac{k}{h} = \lambda$. Thus

$$
\begin{aligned}
v_m^{n+1} &= \frac{1}{2}v_{m+1}^n + \frac{1}{2}v_{m-1}^n - \frac{1}{2}a\lambda v_{m+1}^n + \frac{1}{2}a\lambda v_{m-1}^n \\
&= v_{m+1}^n\left(\frac{1}{2} - a\lambda\right) + v_{m-1}^n\left(\frac{1}{2} + a\lambda\right).
\end{aligned}
$$

We want to show that this scheme is stable if $|a\lambda| \leq 1$. Then

$$
\left|\frac{1}{2} - a\lambda\right| + \left|\frac{1}{2} + a\lambda\right| \leq 1
$$

$$
\Rightarrow |a\lambda| \leq 1
$$

I have abided by the San Diego State University Honor Code.

*Matteo Polimeno*