Assignment IV

M693B

DR. BLOMGREN

APRIL 5, 2018

MATTEO POLIMENO

1 Problem 2.3.2 from Strikverda

1.1 Function

We used the Forward-Time Central-Space scheme for $u_t + u_x = 0$ on the interval [-1,3] for $0 \leq t \leq 4$ for the following set of the initial data:

$$u_0 = \begin{cases} 1 - |x|, & \text{if } |x| \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

for part a, and

$$u_0 = \sin(x)$$

for part b.
As for the boundary conditions, we used

$$u(t, -1) = 0$$

at the left boundary for part a, while

$$u(t, -1) = -sin(1 + t)$$

for part b.
We used

$$v_M^{n+1} = v_{M-1}^{n+1}$$

at the right boundary for both part a and part b.

For Part a, we run the following function

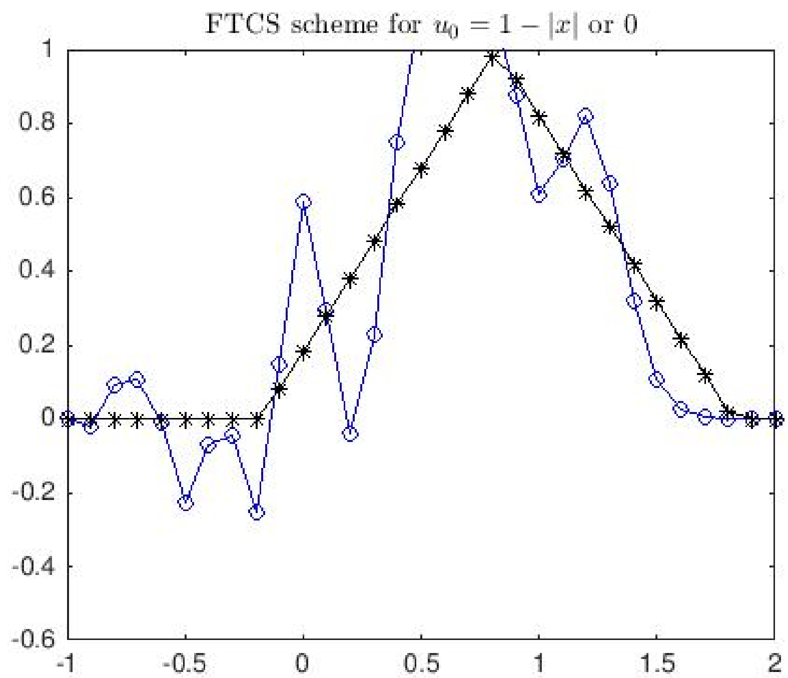
```
%one-way wave equation
function J = hwe(x)

if abs(x) <= 1
    J = 1-abs(x);
else
    J = 0;
end


end
```

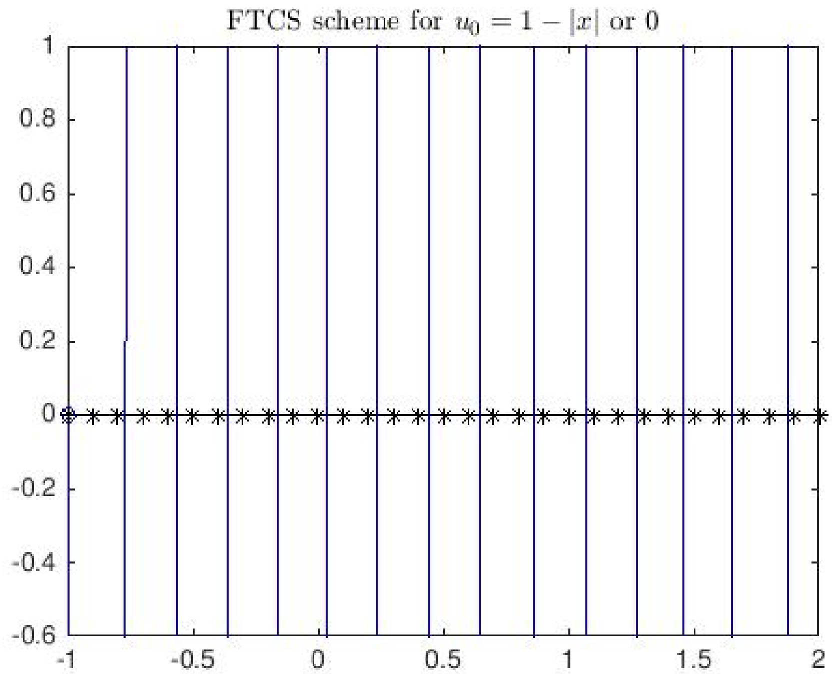
and here is the FTCS scheme

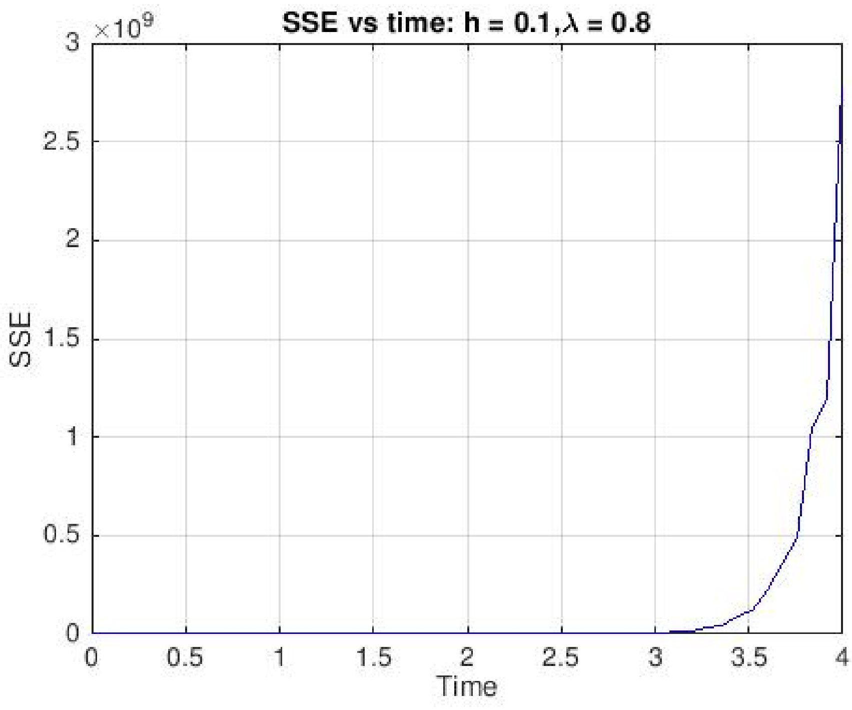
```
%Forward-time central-space scheme
clear all
clc

lambda = .8;
h = 1/10;
x = -1:h:3;
p = length(x);
k = lambda*h;
t = 0:k:4;
q = length(t);


[X,Y] = meshgrid(x,t);

u = zeros(1,p);
for i = 1:p
    if abs(x(i)) <= 1
        u(1,i) = 1 - abs(x(i));
    else
        u(1,i) = 0;
    end
end

%left boundary for part a
for i = 1:q
    u(i,1) = 0;
end

%run the scheme
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((x(j) -  t(i))); %hwe is for 1-abs(x)
    end
end

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


for i = 1:q
    plot(x,u(i,:),'b-o',x,v(i,:),'k-*');
    ylim([-0.6,1])
    xlim([-1,2])
    title('FTCS scheme for $u_{0}=1-|x|$ or $0$' , 'Interpreter', '
        latex')
   M(i) = getframe;
end

for i = 1:q
    E(i,:) = abs((v(i,:)-u(i,:)));
    err(i) = sum(E(i,:)).^2;
end
disp(err(i));

figure()
plot(t,err,'b-')
title(['SSE vs time: h = ' num2str(h) ',\lambda = ' num2str(lambda)])
xlabel('Time')
ylabel('SSE')
grid on
```

We ran the scheme for $h = \frac{1}{10}$, and $\lambda = 0.8$. We plotted the results




FTCS scheme for $u_0 = 1 - |x|$ or 0
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-1
-0.5
0
0.5
1
1.5
2


SSE vs time: h = 0.1,λ = 0.8
$\times 10^9$
3
2.5
2
1.5
1
0.5
0
SSE
0
0.5
1
1.5
2
2.5
3
3.5
4
Time

For Part b, we then ran the following function

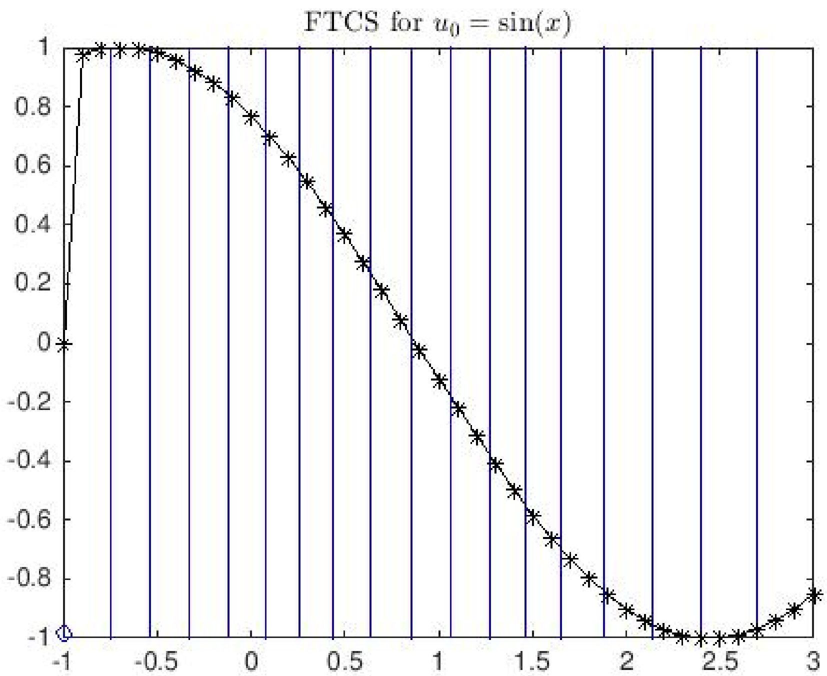
```
%one−way wave equation
function J = hwe(x)

if abs(x) <= 1
    J = 1−abs(x);
else
    J = 0;
end


end
```

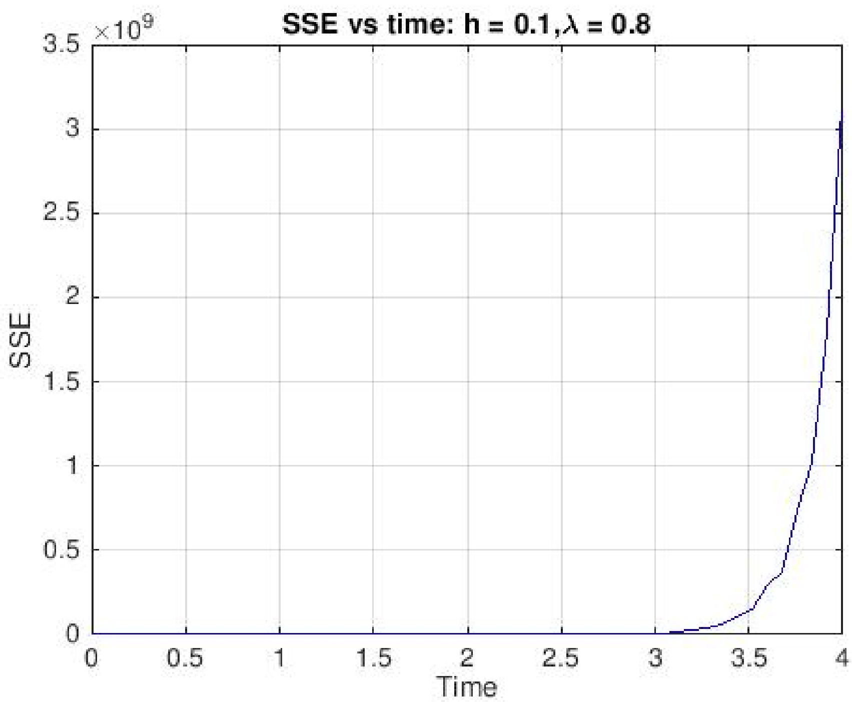
and the FTCS scheme

```
%Forward−time central−space scheme 2.3.2b
clear all
clc

lambda = .8;
h = 1/10;
x = −1:h:3;
p = length(x);
k = lambda*h;
t = 0:k:4;
q = length(t);


[X,Y] = meshgrid(x,t);

u = zeros(1,p);
for i = 1:p
    if abs(x(i)) <= 1
        u(1,i) = 1 − abs(x(i));
    else
        u(1,i) = 0;
    end
end


%left boundary for part b
for i = 1:q
    u(i,1) = −sin(1+i);
end

%run the scheme
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) = hweb((x(j) -  t(i))); %hweb is for sin(x)
        end
end

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

for i = 1:q
        plot(x,u(i,:),'b-o',x,v(i,:),'k-*')
        ylim([-1,1])
        xlim([-1,3])
        title('FTCS for $u_{0}=\sin(x)$', 'Interpreter', 'latex')
        M(i) = getframe;
end

for i = 1:q
        E(i,:) = abs((v(i,:)-u(i,:)));
        err(i) = sum(E(i,:)).^2;
end
disp(err(i));

figure()
plot(t,err,'b-')
title(['SSE vs time: h = ' num2str(h) ',\lambda = ' num2str(lambda)])
xlabel('Time')
ylabel('SSE')
grid on
```

We plotted the results




SSE vs time: h = 0.1,λ = 0.8
×10^9
3.5
3
2.5
2
1.5
1
0.5
0
SSE
0
0.5
1
1.5
2
2.5
3
3.5
4
Time

It can readily be seen that the scheme is unstable for either initial data and they seem to blow up at around the same time, at approximately $t = 3$.

2 Problem 5.3.5 from Strikverda

We used the leapfrog scheme to solve the one-wave equation $u_t + u_x = 0$ on the interval [-1,9] for $0 \le t \le 7.5$. The initial data was given

$$u(0,x) = \begin{cases} \cos \xi_0 x \cos^2(\frac{1}{2}\pi x), & \text{for } |x| - 1 \\ 0, & \text{otherwise} \end{cases},$$

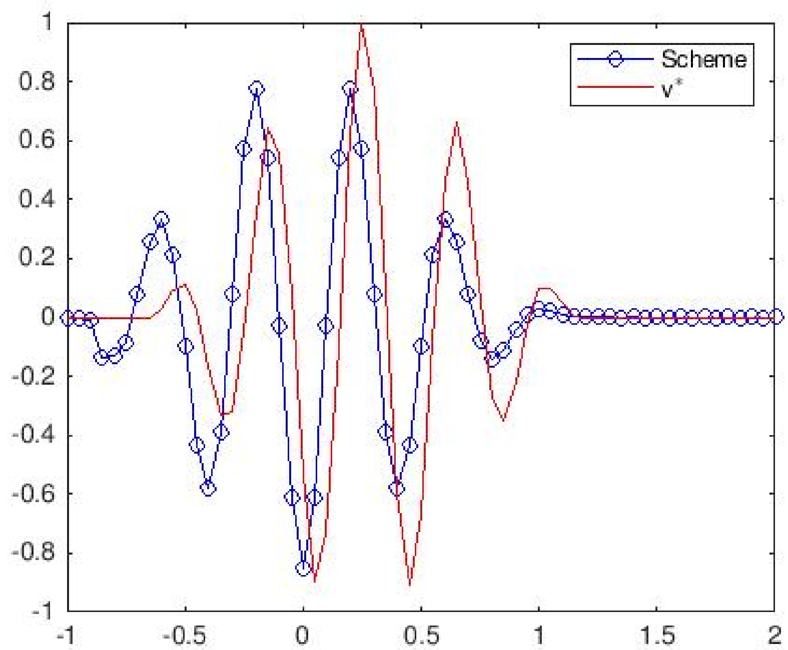
with $\xi_0 = 5\pi$. The grid spacing is 0.05 and $\lambda = 0.95$.

We plotted our results, including an approximate solution computed through the initial data.

```
function J = v_star(x) %solution

csi = 5*pi;
if abs(x) <= 1
    J = cos(csi*x)*cos(0.5*pi*x).^2;
else
    J = 0;
end
end
```

```
clear all
clc

%initialize parameters
l = 0.095;
h = 0.05;
x = -1:h:9;
k = l*h;
t = 0:k:7.5;
m = length(x);
n = length(t);
csi = 5*pi;

%initial data
u = zeros(1,m);
for i = 1:m
    if abs(x(i)) <= 1
        u(1,i) = cos(csi*x(i))*cos(0.5*pi*x(i)).^2;
    else
        u(1,i) = 0;
    end
end

%scheme without dissipation
for i = 1:n-1 %leap frog
    u(i+1,1) = 0; %left boundary condition
    for j = 2:m-2
        u(i+2,j+1) = -l*(u(i+1,j+2) - u(i+1,j)) + u(i,j+1);
        u(i+1,m) = u(i,m-1); %right boundary condition %quasi-characteristic extrapolation
    end
end

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

for i = 1:n
    for j = 1:m
         v(i,j) = v_star((x(j) -  t(i)));
    end
end


for i = 1:n
    plot(x,u(i,:),'b-o',x,v(i,:),'r-')
```

```
        ylim([-1,1])
        xlim([-1,2])
        legend('Scheme','v^{\ast}')
       M(i) = getframe;
end
```

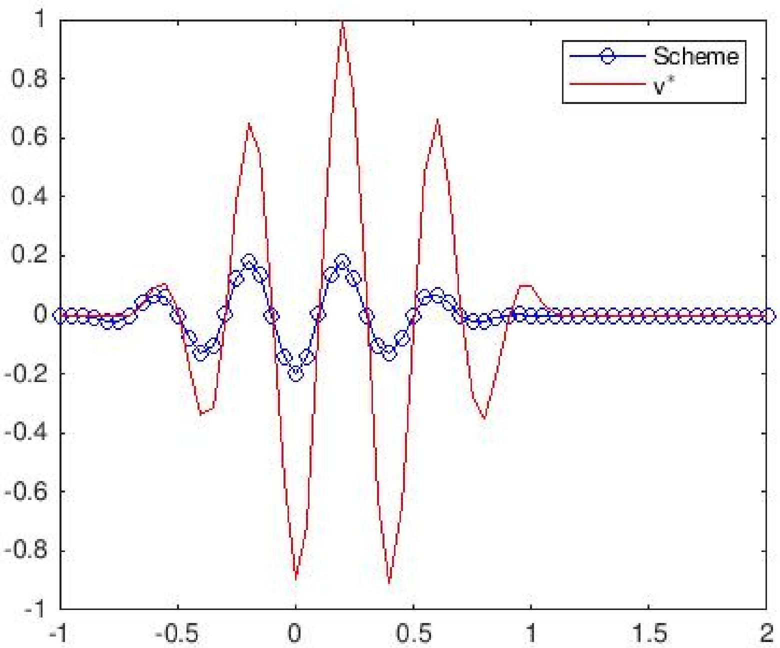


Thus, we see that the scheme seems to travel with the wave packet, at least at the initial stage.

Then, we added a dissipation factor and re-ran the modified leapfrog scheme for a value of dissipation of $\epsilon = 0.5$. Here are the code implemented and the plots obtained

```
clear all
clc

%initialize parameters
l = 0.095;
h = 0.05;
x = -1:h:9;
k = l*h;
t = 0:k:7.5;
m = length(x);
n = length(t);
csi = 5*pi;
```

```
a = 1;
d = 1;
eps = 0.5;

%initial data
u = zeros(1,m);
for i = 1:m
    if abs(x(i)) <= 1
        u(1,i) = cos(csi*x(i))*cos(0.5*pi*x(i)).^2;
    else
        u(1,i) = 0;
    end
end

%scheme with dissipation
for i = 1:n-1
    u(i+1,1) = 0; %left boundary
    for j = 2:m-2
        u(i+2,j+1) = u(i,j+1) - 2*k*a*d*u(i+1,j+1) - (eps*(0.5*h*d).^4)*(u(i,j+1));
        u(i+1,m) = u(i,m-1); %right boundary %quasi-characteristic extrapolation
    end
end

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

for i = 1:n
    for j = 1:m
         v(i,j) = v_star((x(j) -  t(i)));
    end
end


for i = 1:n
    plot(x,u(i,:),'b-o',x,v(i,:),'r-')
    ylim([-1,1])
    xlim([-1,2])
    legend('Scheme','v^{\ast}')
   M(i) = getframe;
end
```



Thus, even though we stopped our movie in a moment where the scheme is losing accuracy, we see that the dispersion has reduced the oscillatory behavior around 0 at the left boundary that was noticeable in the scheme without dissipation.