Assignment III

M693B
DR. BLOMGREN
MARCH 18, 2018

MATTEO POLIMENO

1 Problem 6.3.10 from Strikveda

1.1 Function

Here is the function that we used to plot the solution

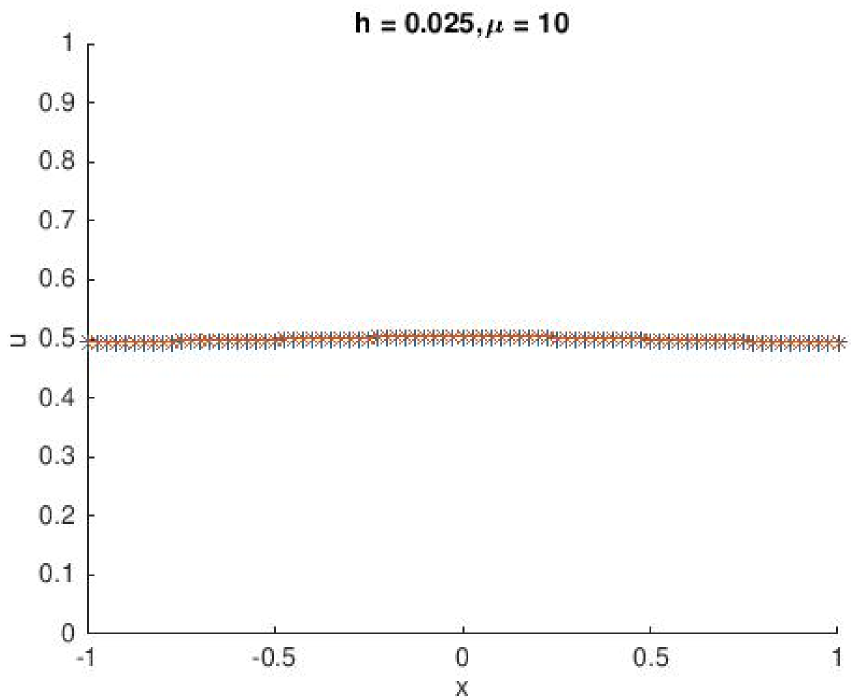
```
function J = ut(t,x,elle)
    J = 1/2;
    for j = 0:elle
        J = J + 2*(-1)^j * ((cos(pi*(2*j+1)*x))/(pi*(2*j+1)))*exp((-pi
            ^2)*(2*j+1)^2*t);
    end

end
```

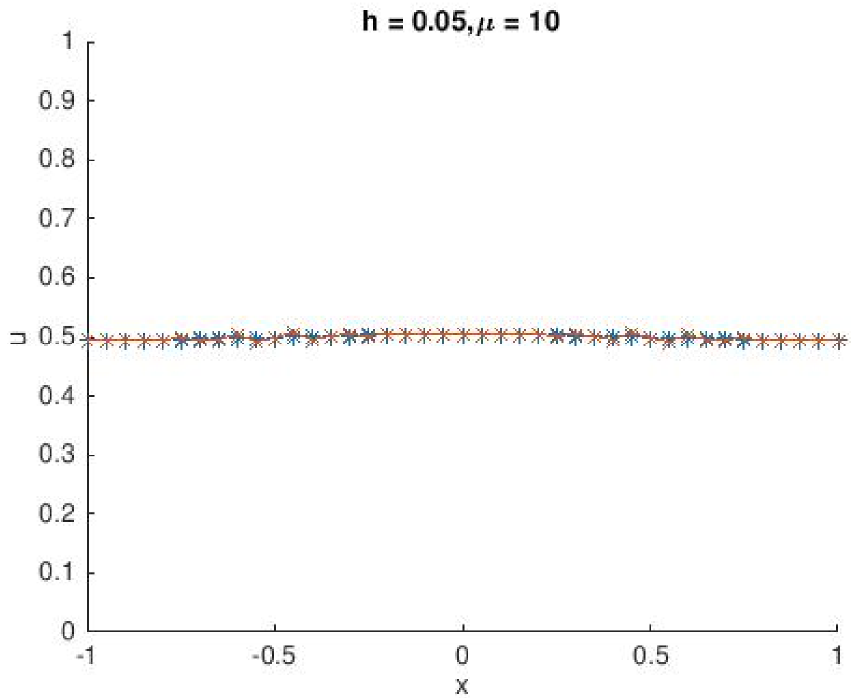
and here is the Crank Nicolson Scheme

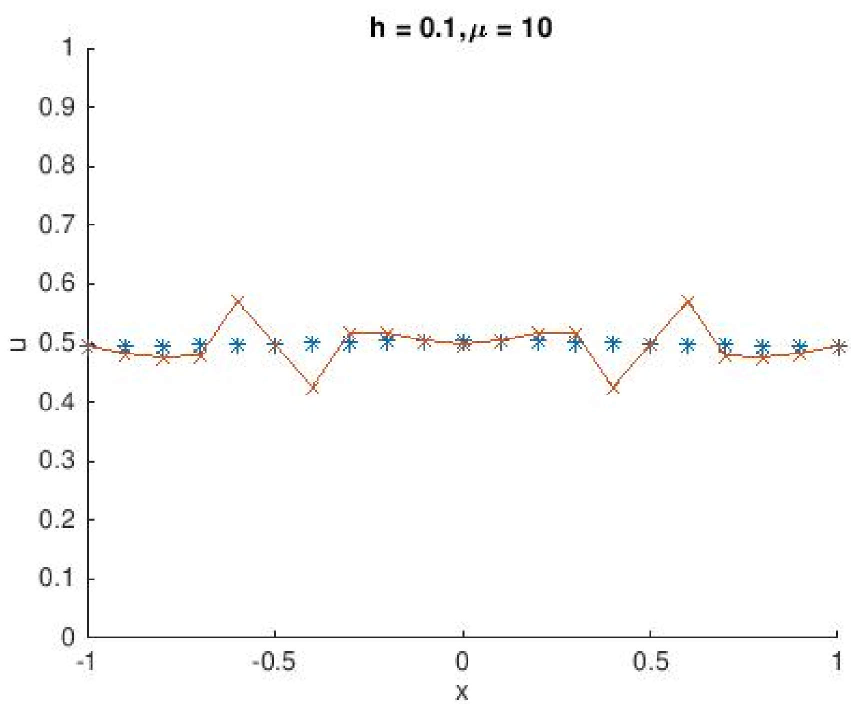
```
clc
clear all

u0 = @(x) (abs(x) < 0.5)+(abs(x) == 0.5)/2;

tmax = 1/2;
elle = 100;
hvals = [1/10 1/20 1/40];

for lambda_switch = 1:2
    for h = hvals
        if lambda_switch == 1
            mu = 1/h; %lambda=mu*h and for lambda=1 we have mu=1/h
        else
            mu = 10;
        end

        k = mu*h^2;
        x = (-1:h:1)';
        m = length(x);
        b = ones(m,1);
        v = zeros(m,1);
        time = 0;

        coeff_matrx_n1 = [(-mu/2)*b (1+mu)*b (-mu/2)*b];
        coeff_matrx_n  = [(mu/2)*b (1-mu)*b (mu/2)*b];

        A = spdiags(coeff_matrx_n1,[-1 0 1], m,m); %make triadiagonal
            matrix %diagonals are at the -1 (meaning 1 down), main and 1
             (meaning 1 up)
        B = spdiags(coeff_matrx_n, [-1 0 1], m,m);
```

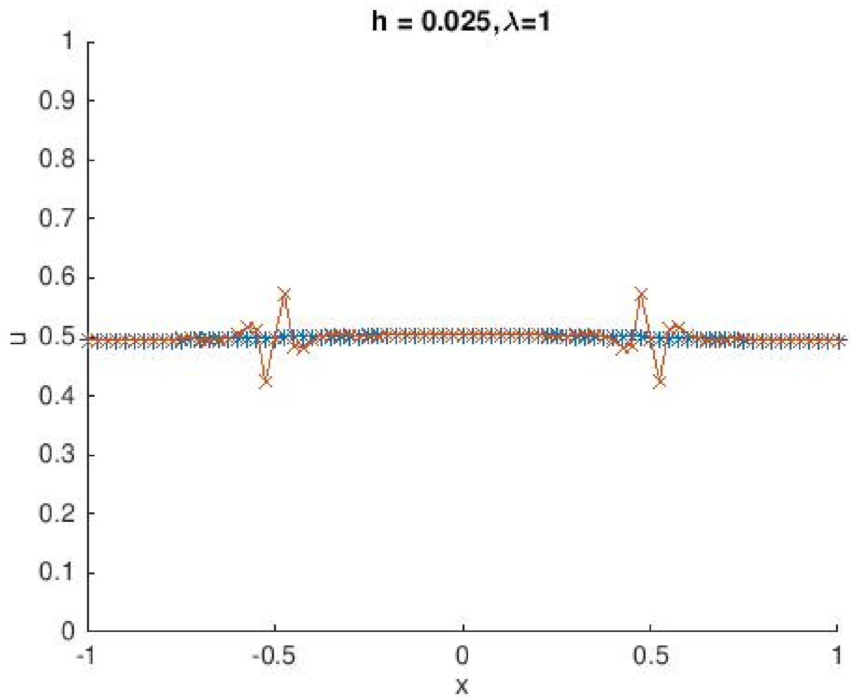
```
        A(1,2) = 0;
        A(m,m-1) = 0;

        v(1) = ut(0,1,elle);
        for M = 2:m-1
            v(M) = u0(x(M)); %initial data
        end
        v(m) = ut(0,m,elle);

        while time < tmax
            time = time+k; %compute time

            A(1,1) = ((1-mu)*v(1)+mu/2*v(2))/ut(time,1,elle);
            A(m,m) = (mu/2*v(m-1)+(1-mu)*v(m))/ut(time,m,elle);
            v = A\(B*v);
        end

        uxsol = ut(time,x,elle);
        figure;
        hold on
        if lambda_switch == 1
            type = '\lambda=1';
        else
            type = '\mu = 10';
        end
        title(['h = ' num2str(h) ',' type]);
        plot(x,uxsol,'*',x,v,'x-');
        xlabel('x')
        ylabel('u')
        axis([-1 1 0 1])
        hold off;
        if lambda_switch == 1
            disp(['h: ' num2str(h)]);
            disp(['Supremum Norm: ' num2str(max(abs(uxsol-v)))]);
            disp(['L2 Norm: ' num2str(norm(uxsol-v))]);
        end
    end
end
```

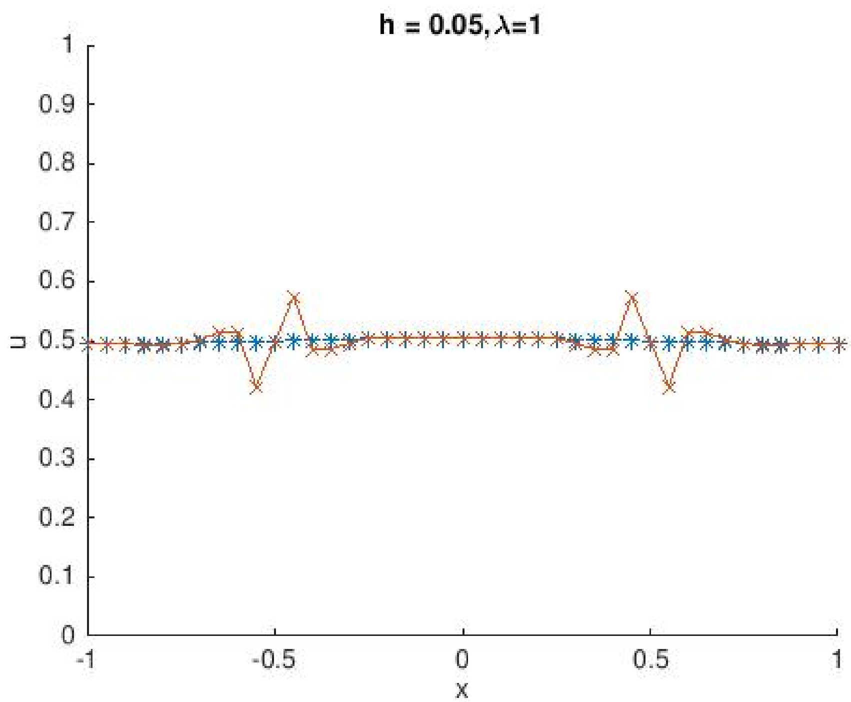
We ran the scheme for $h = \frac{1}{10}$, $h = \frac{1}{20}$ and $h = \frac{1}{40}$ and for $\lambda = 1$ and $\mu = 10$. We plotted the results

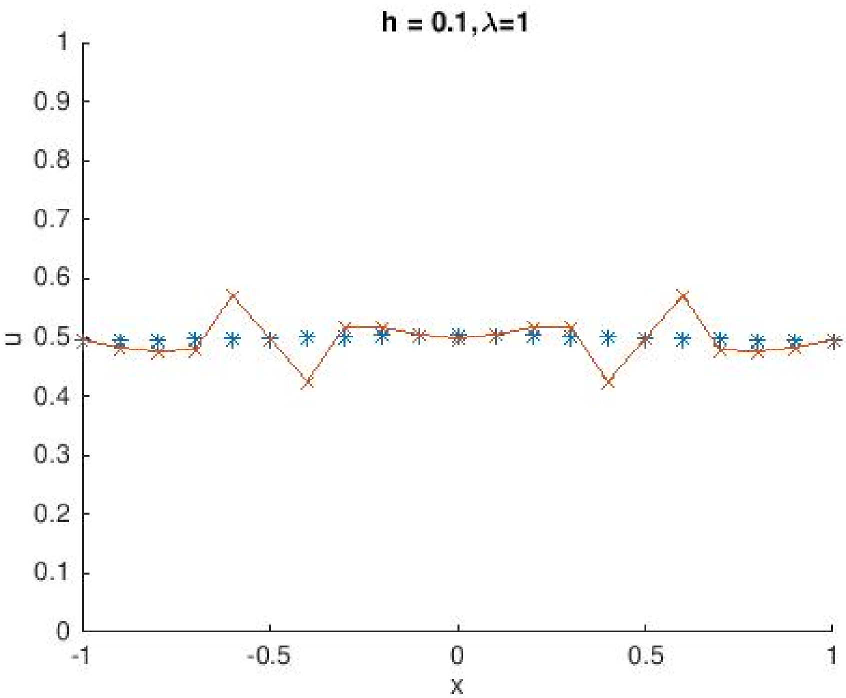



h = 0.05, μ = 10
u
x
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-1
-0.5
0
0.5
1


h = 0.1, μ = 10
u
x
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-1
-0.5
0
0.5
1


h = 0.025, λ=1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
u
-1
-0.5
0
0.5
1
x


h = 0.05, λ=1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
u
-1
-0.5
0
0.5
1
x


h = 0.1, λ=1
u
x
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-1
-0.5
0
0.5
1

It can readily be seen that the scheme is extremely accurate for $\mu = 10$ and $h = 1/40$, and for such valueof μ its usefulness decreases as h increases, still being pretty useful for $h = 1/20$, but becoming useless for $h = 1/10$.

As for the scheme with the given λ value of 1,we readily see that it does not matches the solution at the initial data $|x| = \frac{1}{2}$ for any value of h and its usefulness decreases as h increases.

As for Supremum norm and the L^2 norm, we obtained the following results:

$$
\begin{aligned}
&For\ h = 0.1 : \\
&Supremum\ Norm : 0.076479, \\
&L2\ Norm : 0.15861; \\
\\
&For\ h = 0.05 : \\
&Supremum\ Norm : 0.075421, \\
&L2\ Norm : 0.15828; \\
\\
&For\ h = 0.025 : \\
&Supremum\ Norm : 0.075401, \\
&L2\ Norm : 0.15845;
\end{aligned}
$$