MIDDLE EAST TECHNICAL UNIVERSITY
MATHEMATICS DEPARTMENT

MATH480

NUMERICAL METHODS FOR DIFFERENTIAL EQUATIONS

PROF. DR. MÜNEVVER TEZER

A MATLAB IMPLEMENTATION FOR 1D WAVE EQUATION

30.06.2021

GÜRAY ÖZGÜR 2167054

1 PROBLEM DEFINITION

1.1 WAVE EQUATION

The general formula for the one dimensional wave equation with its boundary and initial conditions is given in Equation 1.

$$
\begin{aligned}
\frac{\partial^2 u}{\partial t^2} &= \alpha^2 \frac{\partial^2 u}{\partial x^2} && 0 < x < L,\ 0 < t < T \\
u(0,t) = u(L,t) &= 0 && t > 0 \\
u(x,0) &= f(x) && 0 \le x \le L \\
\frac{\partial u}{\partial t}(x,0) &= g(x) && 0 \le x \le L
\end{aligned}
\tag{1}
$$

where $f(x)$ and $g(x)$ are given functions, α is a constant.

Here, in our case, the followings are specified as shown in Equation 2 and a numerical solution is needed.

$$
\begin{aligned}
\alpha &= 1 \\
L &= 1 \\
T &= 0.3 \\
f(x) &= \frac{1}{2}x(1-x) \\
g(x) &= 0
\end{aligned}
\tag{2}
$$

1.2 DISCRETIZATION

For a numerical solution, a discretization procedure is needed. For both the time and the space, it is required to discretize the intervals. Assume that the interval $[0, L]$ is divided to M steps and the interval $[0, T]$ is divided to N steps with the same step sizes. Let the step sizes be h and k respectively. Then, the discretization can be written as

$$x_i = ih, \qquad i = 0, 1, \ldots, M$$

where $h = L/M$.

$$t_j = jk, \qquad j = 0, 1, \ldots, N$$

where $k = T/N$.

1.3 DERIVATION OF FINITE DIFFERENCE APPROXIMATION

The numerical solution will be obtained from the finite difference method. Finite difference approximation of the wave equation is obtained by using central differences for both u_{xx} and u_{tt}, which are shown in Equations 3 and 4. The wave equation using these central differences is shown in Equation 5, where the truncation error is $O(h^2 + k^2)$ is not shown.

$$u_{xx} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \tag{3}$$

$$u_{tt} = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} \tag{4}$$

$$\frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} = \alpha^2 \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \tag{5}$$

By rearranging Equation 5, it is obtained that

$$u_{i,j+1} = m^2 u_{i-1,j} + 2(1 - m^2)u_{i,j} + m^2 u_{i+1,j} - u_{i,j-1} \tag{6}$$

where $m = \alpha k/h < 1$ for $i = 1, \dots, M-1$ and $j = 1, \dots, N-1$.

At first time step, an out of region value is obtained and it is handled by initial conditions, i.e. by approximating the time derivative with central difference. Here, Equation 7 is obtained.

$$\frac{\partial u}{\partial t}(x_i, 0) = \frac{u_{i,1} - u_{i,-1}}{2k} = g(x_i) \tag{7}$$

Since $g(x_i) = 0$ in our problem, $u_{i,-1} = u_{i,1}$ is obtained for the out of region value.

By substituting $u_{i,-1} = u_{i,1}$ to Equation 6, the form of the equation for the first time step is obtained and shown in Equation 8.

$$u_{i,1} = \frac{m^2}{2} u_{i-1,0} + (1 - m^2)u_{i,0} + \frac{m^2}{2} u_{i+1,0} \tag{8}$$

1.3.1 MATRIX-VECTOR FORM

A matrix-vector form of finite difference approximation can be obtained Equations 6 and 8 for the wave equation, which are written again for convenience.

$$u_{i,1} = \frac{m^2}{2} u_{i-1,0} + (1 - m^2)u_{i,0} + \frac{m^2}{2} u_{i+1,0} \tag{9}$$

$$u_{i,j+1} = m^2 u_{i-1,j} + 2(1-m^2)u_{i,j} + m^2 u_{i+1,j} - u_{i,j-1} \tag{10}$$

where $m = \alpha k/h < 1$ for $i = 1, \ldots, M-1$ and $j = 1, \ldots, N-1$.

By Equations 9 and 10, matrix-vector forms can be obtained, which are shown Equations 11 and 12 respectively.

For $j = 1, \ldots, N-1$,

$$U^{(j+1)} = AU^{(j)} - U^{(j-1)} \tag{11}$$

where

$$U^{(j)} = \begin{bmatrix} u_{1,j} \\ u_{2,j} \\ u_{3,j} \\ \vdots \\ u_{M-1,j} \end{bmatrix},$$

$$A = \begin{bmatrix} 2(1-m^2) & m^2 & & & & \\ m^2 & 2(1-m^2) & m^2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & m^2 & 2(1-m^2) & m^2 \\ & & & & m^2 & 2(1-m^2) \end{bmatrix}.$$

For $j = 0$,

$$U^{(1)} = BU^{(0)} \tag{12}$$

where $U^{(j)}$ is same and $B = A/2$ as in Equation 11.

Note that in Equation 12, an additional term coming from the initial conditions are eliminated not only because $g(x) = 0$ but also because $f(0) = f(1) = 0$, which is a special situation in this specified question.

2 NUMERICAL RESULTS

For different time steps, the numerical solution is tested. Specified values are used as in stated Equation 2. The MATLAB code can be found in Appendix A, and also a zipped version is submitted through ODTUclass.

Exact solution is written in MATLAB code in order to compare the numerical solution, and an approximation of the exact solution is calculated with a small truncation error, as 1000 iterations are done to calculate the exact solution.

For time step $k = 0.05$ is chosen for comparison in table format, since the dimensions are quite small and can be written. In Table 1, numerical solution is tabulated and in Table 2, analytical solution is tabulated. When compared, the difference is seen quite small. To compute an error, MSE is chosen and calculated, whose value is found as $4.3937x10^{-5}$. Furthermore, a visualization is better, thus for $k = 0.05$ case, numerical and analytical solutions are shown in Figures 1 and 2.

Table 1: Numerical solutions for $h = 0.1$ and $k = 0.05$

u	**t = 0.05**	**t = 0.1**	**t = 0.15**	**t = 0.2**	**t = 0.25**
x = 0.1	0.0450	0.0438	0.0403	0.0355	0.0301
x = 0.2	0.0800	0.0788	0.0750	0.0688	0.0605
x = 0.3	0.1050	0.1038	0.1000	0.0938	0.0850
x = 0.4	0.1200	0.1187	0.1150	0.1088	0.1000
x = 0.5	0.1250	0.1237	0.1200	0.1137	0.1050
x = 0.6	0.1200	0.1187	0.1150	0.1087	0.1000
x = 0.7	0.1050	0.1038	0.1000	0.0938	0.0850
x = 0.8	0.0800	0.0788	0.0750	0.0688	0.0605
x = 0.9	0.0450	0.0437	0.0403	0.0355	0.0301

Table 2: Analytical solutions for $h = 0.1$ and $k = 0.05$

y	**t = 0.05**	**t = 0.1**	**t = 0.15**	**t = 0.2**	**t = 0.25**
x = 0.1	0.0437	0.0400	0.0350	0.0300	0.0250
x = 0.2	0.0787	0.0750	0.0687	0.0600	0.0500
x = 0.3	0.1037	0.1000	0.0937	0.0850	0.0737
x = 0.4	0.1187	0.1150	0.1087	0.1000	0.0887
x = 0.5	0.1237	0.1200	0.1137	0.1050	0.0937
x = 0.6	0.1187	0.1150	0.1087	0.1000	0.0887
x = 0.7	0.1037	0.1000	0.0937	0.0850	0.0737
x = 0.8	0.0787	0.0750	0.0687	0.0600	0.0500
x = 0.9	0.0437	0.0400	0.0350	0.0300	0.0250

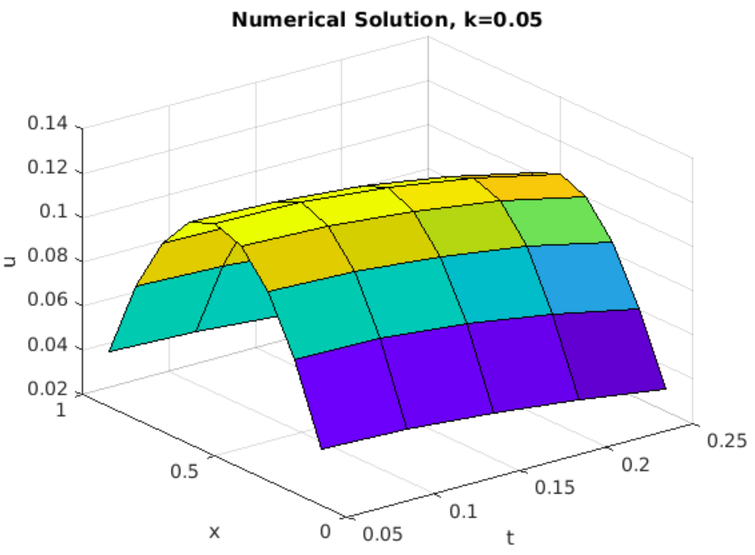

Figure 1: Numerical solution when $k = 0.05$

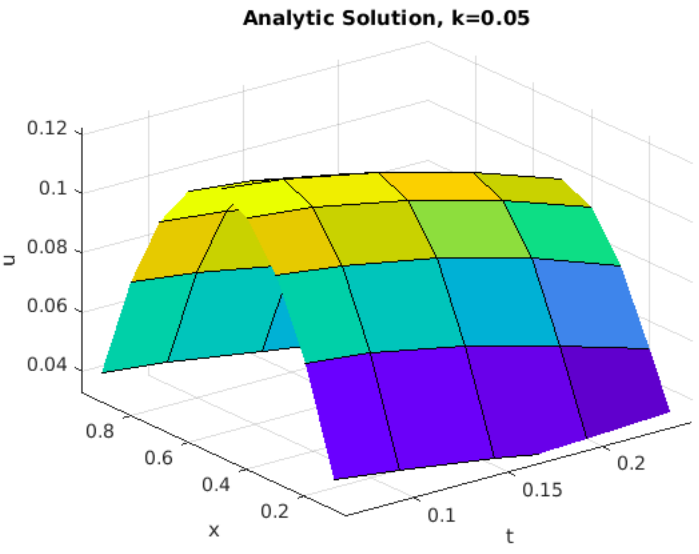

Figure 2: Analytical solution when $k = 0.05$

To see the effect of step size, time steps are adjusted to $k = 0.01$ and $k = 0.005$, results are shown in Figures 3, 4, 5, and 6. When step size decreases, the error also decreases. However, computational complexity increases, thus, it takes more time. It can be said that there is a trade-off between accuracy and speed. Moreover, MSE values are calculated in each case, and tabulated in Table 3. As can be seen from Table 3, decreasing step size results in decrease in error.

Table 3: MSE between numerical and analytical solution for different time steps

k	***0.05***	***0.01***	***0.005***
MSE	$4.3937x10^{-5}$	$2.7293x10^{-6}$	$8.8816x10^{-7}$

To test the stability, when $k = 0.05$, step size for distance has changed to $h = 0.01$. In this case, since $m > 1$, it is expected to observe stability, which is seen in Figures 7 and 8. Also, for the case, MSE value was as high as 52.1499.

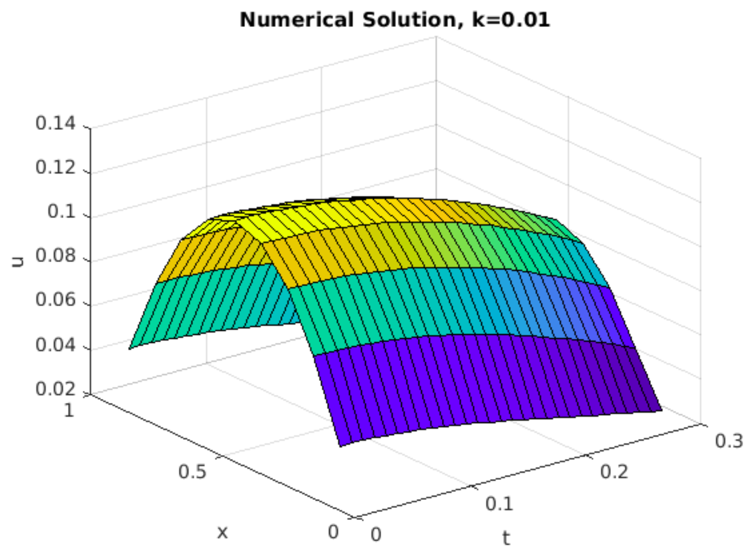

Figure 3: Numerical solution when $k = 0.01$

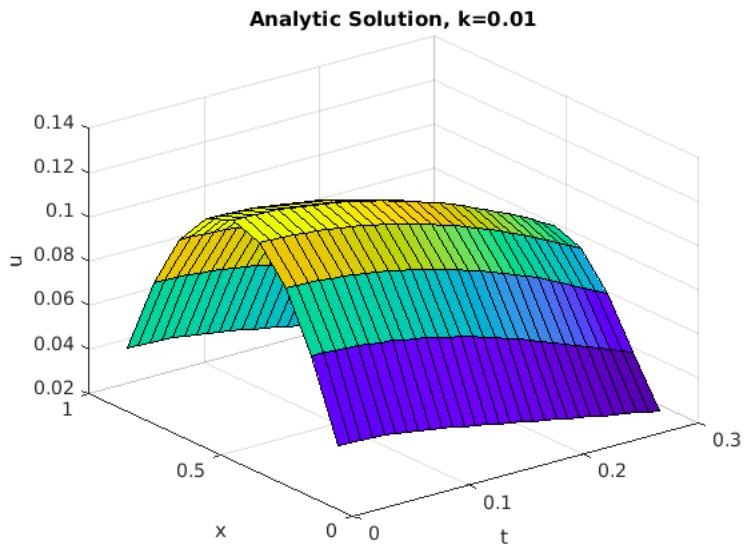

Figure 4: Analytical solution when $k = 0.01$

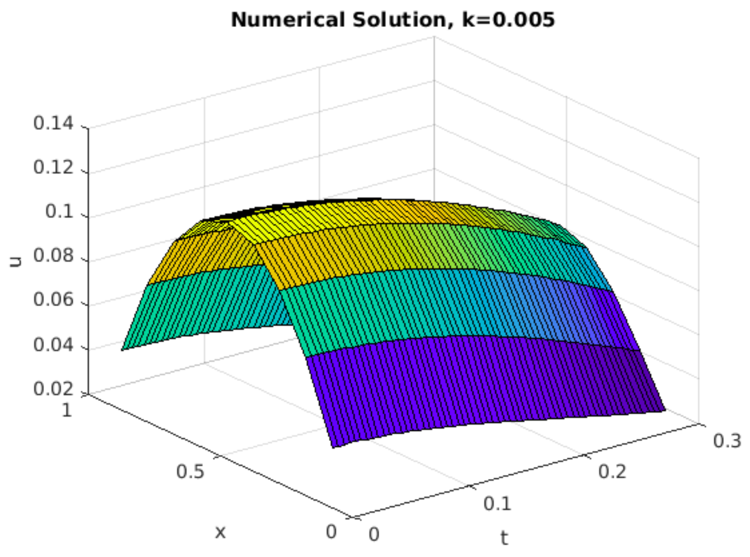

Figure 5: Numerical solution when $k = 0.005$

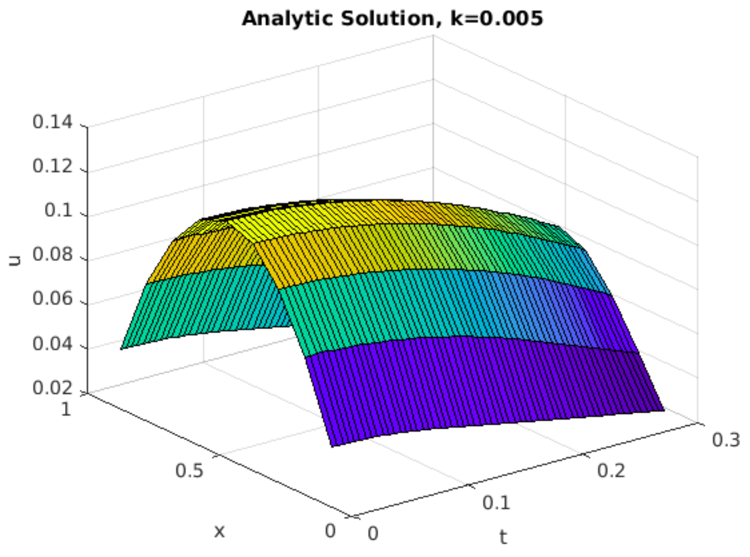

Figure 6: Analytical solution when $k = 0.005$

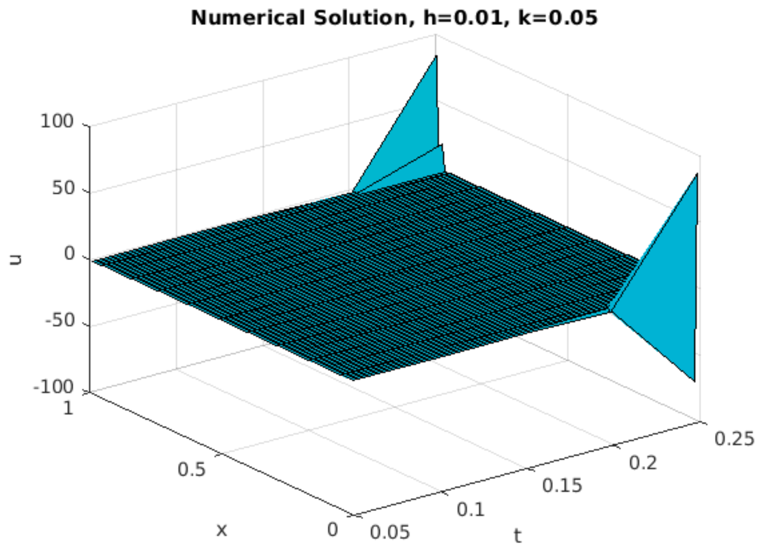

Figure 7: Numerical solution when $h = 0.01$ and $k = 0.05$

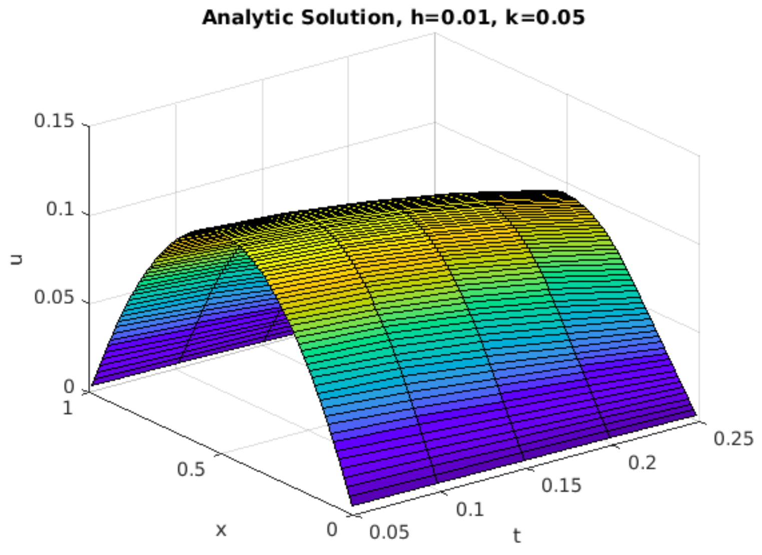

Figure 8: Analytical solution when $h = 0.01$ and $k = 0.05$

3 APPENDICES

A MATLAB Code

```matlab
% Initialization
L = 1;
h = 0.0 1;
T = 0.3;
k = 0.05;
m = k/h;

x = [0:h:L];
t = [0:k:T];
x = x(2:end-1);
t = t(2:end-1);
M = length(x);
N = length(t);

% Matrix Definition for Matrix-Vector Form
a = 1-m*m;
b = m*m;
A = diag(a*ones(1,M)) + diag(b*ones(1,M-1)/2,1) + diag(b*ones(1,M
    -1)/2,-1);

% Numeric Solution
u = zeros(M,N);
u(:,1) = x.*(1-x)/2;
j = 1;
u(:,j+1) = A*u(:,j);
for j = 2:N-1
    u(:,j+1) = 2*A*u(:,j) - u(:,j-1);
end

% Analytic Solution
[t,x] = meshgrid(t,x);
y = 0;
for n = 1:1000
    y = y + 1/(n^3)*(1-cos(n*pi)).*cos(n*pi*t).*sin(n*pi*x);
end
y = (2/pi^3)*y;

D = abs(y-u).^2;
MSE = sum(D(:))/numel(y)
```

```matlab

% Plotting
figure
surf(t,x,u)
title(['Numerical Solution, h=', num2str(h), ', k=', num2str(k)])
xlabel('t')
ylabel('x')
zlabel('u')

figure
surf(t,x,y)
title(['Analytic Solution, h=', num2str(h), ', k=', num2str(k)])
xlabel('t')
ylabel('x')
zlabel('u')
```