Table of Contents

1D One Way Wave Equation 1
Initial Given values 1
Calculating Numerical and Analytical Solution 1
Initial Condition 2
FTCS Scheme 2
Leap Frog Scheme 2
Analytical Solution 3
Accuracy Test 3
Plot 3
Discussion 4

1D One Way Wave Equation

```
% Solving 1D One Way Wave Equation using Leap Frog Scheme
% Author: Pradeep Singh
% Date: 10/30/2017

clc
close all
```

Initial Given values

```
% Domain values for x and t
x_min = -1;
x_max = 1;
t_min = 0;
t_max = 1.2;
c = 1;      % c = 1 (given value)

k =10;  % Number of times we double our grid size
        % for accuracy test.
m = 2;  % Resolution
n = 2;  % Calculate accuracy at this given point

% Store data in store matrix to calculate the accuracy.
% First row will hold analytical solution and the
% Second row will hold numerical solution (FTBS Scheme).

store = zeros(2, k);
```

Calculating Numerical and Analytical Solution

```
% Loop k times
for j=1:k

    if(j > 1)
        m = 2*m;  %doubles our resolution
```

```
end

%initialize values and grids
x_grid = m;
t_grid = m;

%spatial and temporal step sizes
dx = (x_max-x_min)/(x_grid-1);
dt = (t_max-t_min)/(t_grid-1);
```

Initial Condition

```
% Initial Condition
% u(x,0) = u0(x) = sin(2*pi*x)
syms x
f(x) = sin(2*x*pi);
Initial_cond = zeros(x_grid, 1);

% CFL condition
CFL = c*dt/dx; % c =1 (given)

% Cal values for 1st row using IC
Initial_cond(1, 1) = f(x_min);
for i=1:x_grid-1
   Initial_cond(i+1, 1) = f(i*dx+x_min);
end
```

FTCS Scheme

```
% FTBS coefficient Matrix
% Multiply by 1/2 to convert into FTCS
U = CFL*eye(x_grid) - (1/2)*CFL*diag(ones(x_grid-1, 1), 1)+...
    (1/2)*CFL*diag(ones(x_grid-1, 1), -1);

% Boundary values for FTCS coefficient Matrix
U(1, x_grid) = (1/2)*CFL;
U(x_grid, 1) = -(1/2)*CFL;

% For time step 1, we have IC
U_old = Initial_cond(:, 1); %timestep 1

% For second time step, use FTCS scheme
U_new = U*U_old; %timestep 2
```

Leap Frog Scheme

```
   % Calculating Leapfrog matrix
   P = zeros(x_grid, x_grid);
   Leap = P + CFL*diag(ones(x_grid-1, 1), -1)-CFL*diag(ones(x_grid-1,
1), 1);

   % Boudary condition for Leap frog matrix
```

```
    Leap(1, 2) = -CFL;
    Leap(x_grid, x_grid-1) = CFL;
    Leap(1, x_grid) = CFL;
    Leap(x_grid, 1) = -CFL;

    % Leap frog scheme, starting from 3 time step
    for i=3:t_grid
        U_next = Leap*U_new+U_old;
        U_old = U_new;
        U_new = U_next;
    end
```

Analytical Solution

```
    % Cal the Analytic Sol at t =1.2
    Analytical = zeros(1, x_grid);
    t = 1.2;
    Analytical(1, 1) = f(x_min-t);

    % Calculating analytical solution using function f
    % and looping x_Grid-1 times.
    for i=1:x_grid-1
        Analytical(1, i+1) = f((i*dx+x_min)-t);
    end

    % Storing data in store matrix for calculating accuracy
    store(1, j) = Analytical(1, n+(i-1));
    store(2, j) = U_new(n+(i-1), 1);

end
```

Accuracy Test

```
% Computing the order of accuracy
accuracy = zeros(1, k-1);
for i=1:(k-1)
   accuracy(i) = log2(abs(((store(1,i)-store(2,i)))/(store(1, i+1)-
store(2, i+1))));
end
```

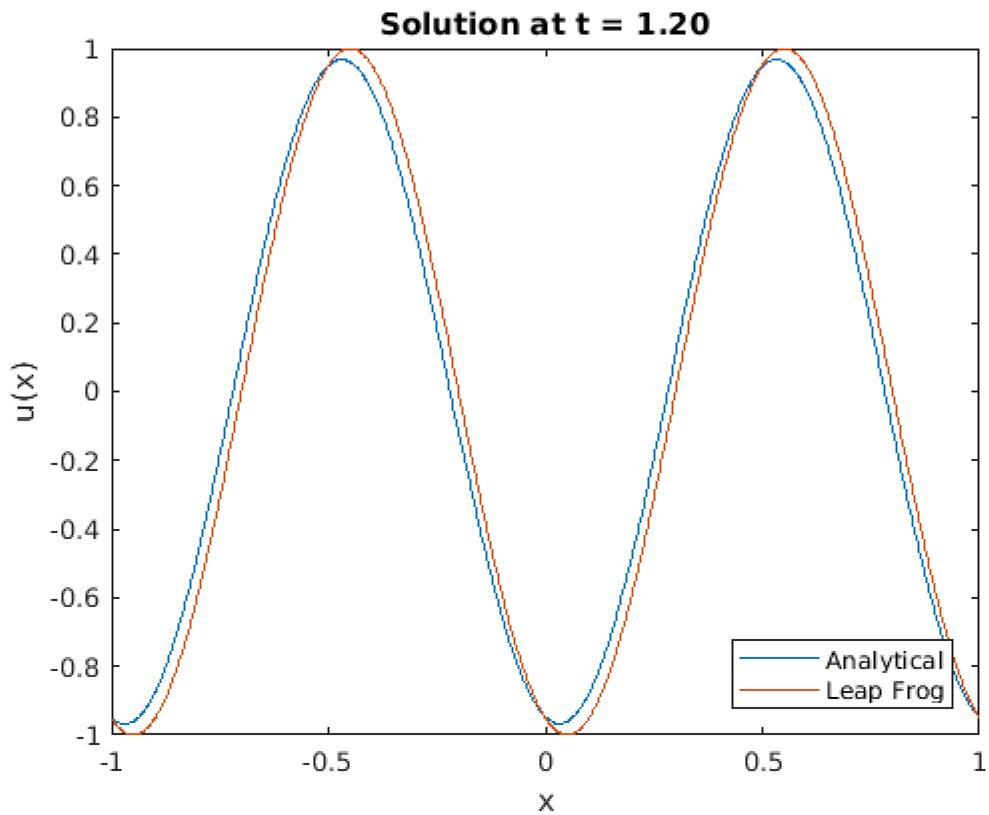
Plot

```
% Plot the analytical and numerical solution
x = linspace(x_max, x_min, x_grid);

plot(x, U_next');
hold on
plot(x, Analytical);
hold off

axis([-1 1 -1 1])
str = sprintf('Solution at t = %.2f', 1.2);
```

```
title(str)
xlabel('x')
ylabel('u(x)')
legend('Analytical','Leap Frog', 'Location', 'Southeast');
```



Discussion

```
% As we increas k, our grid size decrease and we get much finer and
 accurate plots.
% Also, with increase in k, we get accuracy close to 2.

% Eg: With k = 10, our order of accuracy is 1.99, which is very close
 to 2.
```

Published with MATLAB® R2017a