

Computational Electromagnetics and Applications
Professor Krish Sankaran
Indian Institute of Technology Bombay
Exercise 20
Finite Volume Time Domain Method-III

We are now going to look into a very simple straight forward problem of using finite volume method we have discussed it length about the method and its formulation for Maxwell equation it has been very well tried and tested for various applications and I have shown you some of the applications for the course of the lectures we looked at the spiral antenna simulation you have looked into waveguide simulation we have also look into perfectly match layer applications in finite volume method so now using a Matlab example we want to see how the simple problem as such as hyperbolic problem can be formulated and we can test one of the problems that we normally come across hyperbola formulation which is heat collection problem and see how this problem can be model using finite volume method. So let's look into the basic equation which we are interested in which is the hyperbolic equation (Refer Slide Time: 01: 17)

The image shows a handwritten derivation on a whiteboard. At the top, the general hyperbolic equation is written as $\alpha \frac{\partial u}{\partial t} + \nabla \cdot F(u) = 0$. Below this, it is simplified to $\frac{\partial u}{\partial t} + \frac{\partial \psi}{\partial x} = 0$, with a note $F(u) = \psi$. The next step shows the finite volume formulation: $\int_{x_a}^{x_b} \frac{\partial u}{\partial t} dx = - \int_{x_a}^{x_b} \frac{\partial \psi}{\partial x} dx$. This is then simplified to $= - \psi [u(x_b) - u(x_a)]$. An NPTEL logo is visible in the bottom left corner of the whiteboard image.

So the hyperbolic equation in general will have the form $\alpha \frac{\partial u}{\partial t} + \nabla \cdot F(u) = 0$. And of course there will be a parameter Alpha that is a parameter that we multiply this is general general formulation of the hyperbolic form but let's take the conservation law and Maxwell equation can be casted in this particular form but let's take a simple one dimensional form applied for simulating a problem on matlab so let's start with problem where we have only one dimension so I will have $\frac{\partial u}{\partial t} + \frac{\partial \psi}{\partial x} = 0$ so there is a relationship between these two terms so this is a flux function this the divergence all the flux

function where the flux function $F(u)$ is equal to see once you have this week and write them in the integral form integrating along the domain you integrate it along x so X goes from x of a to x of b we have $\int_a^b \frac{\partial u}{\partial t} dx = -\psi [u(x_b) - u(x_a)]$ so we have a minus sign that is coming which is $-$ of the flux function integration so integral of x a to x b $\frac{\partial u}{\partial t} dx = -\psi [u(x_b) - u(x_a)]$ this can be simplified into is equal to minus sign outside will take $c[u(x_b) - u(x_a)]$ so this is the value so now we are going to use the midpoint rule so we can apply certain algorithm for the left hand side come as well so let's take it further.

(Refer Slide Time: 04: 10)

The image shows a handwritten derivation on a whiteboard. At the top, the integral equation is written: $\int_{x_a}^{x_b} \frac{\partial u}{\partial t} dx = -\psi [u(x_b) - u(x_a)]$. Below this, the left-hand side is simplified to $(x_b - x_a) \frac{\partial u^*}{\partial t} =$. The interval $[x_a, x_b]$ is then expressed as $[(i - 1/2)\Delta x, (i + 1/2)\Delta x]$, with a boxed note $i\Delta x = x_b$. A number line diagram below shows points $(i - 1/2)$, i , $(i + 1/2)$, $i + 1$, and $(i + 3/2)$. Above i is the label u , and above $(i + 1/2)$ is the label u^* . A box on the left contains the approximation $u \approx u^*$. The NPTEL logo is visible in the bottom left corner.

So what we have got here is integral X_A to X_B $\frac{\partial u}{\partial t} dx = -\psi [u(x_b) - u(x_a)]$ and this particular left hand side term can be simplified further by using the midpoint rule and the midpoint rule will give us $(x_b - x_a) \frac{\partial u}{\partial t} =$ is equal to the same thing that we have on the right hand side which is the control volume such that so if we choose the control volume $i - \text{half } \Delta x$ comma $i + \text{half } \Delta x$ so this is a control volume so if we take a line like this and let's say this is a minus half this is a plus half this is a $+ 3$ by 2 so on and so forth the control volume i is going to be centred exactly at i and $i + 1$ so on and so forth so these are the central control volumes.

And where so if we say this is the control volume and we have the centre of the control volume at $a \Delta x$ Δx is equal to h we say $i \times h$ so this is the way we structure the control volume control volumes will have the value stored at the centre here and we can approximate the value at the edges using a very simple approximation that the value used r r here is going to be the value at the centre itself Sophie say you approximately equal to you star so this is the approximation that we are using and once we do that we can use this information to compute the right hand side this is the information that we will use to compute

the flux function where as we will keep you as you itself on the left hand side we'll see how you do that step by step so what we have is.

(Refer Slide Time: 07: 47)

The image shows a whiteboard with handwritten mathematical equations. The top equation is $(x_b - x_a) \frac{\partial u}{\partial t} = - \{ \psi(u(x_b)) - u(x_a) \}$. Below it, $(x_b - x_a)$ is annotated with $\Delta x = h$ and $u^* \approx u$. The middle equation is $\frac{u_i^{n+1} - u_i^n}{\Delta t} = -\frac{1}{h} [\psi(x_{i+1/2}) - \psi(x_{i-1/2})]$. The bottom equation is $u_i^{n+1} = u_i^n + \left(-\frac{\Delta t}{h}\right) [\psi(x_{i+1/2}) - \psi(x_{i-1/2})]$. An NPTEL logo is visible in the bottom left corner of the whiteboard.

$(x_b - x_a)$ multiplied by $\frac{\partial u}{\partial t}$ is equal to minus $\{ \psi(u(x_b)) - u(x_a) \}$ for this we are going to use approximation you star is approximately equal to $u(x_a - x_b)$ will be given by Δx which is equal to h so when we rearrange the term and use the forward differencing for time derivative what we get is $u_i^{n+1} - u_i^n$ and divided by Δt is equal to minus $\frac{1}{h}$ so I have taken the h on the other side x the flux function at $x(i + \frac{1}{2})$ minus the flux function defined at $x(i - \frac{1}{2})$ so I have to close the bracket so this if I rearrange it little bit what I will get is $u_i^{n+1} = u_i^n + \left(-\frac{\Delta t}{h}\right) [\psi(x_{i+1/2}) - \psi(x_{i-1/2})]$ and for computing the value at $\psi(x_{i+1/2})$ and $(x - \frac{1}{2})$.

(Refer Slide Time: 10: 19)

$$\int_{x_a}^{x_b} \frac{\partial u}{\partial t} dx = -\psi [u(x_b) - u(x_a)]$$

$$(x_b - x_a) \frac{\partial u^*}{\partial t} =$$

$$\left[(i - \frac{1}{2})\Delta x, (i + \frac{1}{2})\Delta x \right] \quad \boxed{i\Delta x = \Delta x}$$

$$\boxed{u \approx u^*}$$

$\frac{u}{(i - \frac{1}{2}) \quad i \quad (i + \frac{1}{2}) \quad i+1 \quad (i + \frac{3}{2})}$

I use this approximation which is the flux the field value at this point at (i plus half) and (i minus half) is going to be given by the value of the centre itself so this is a very crude approximation

(Refer Slide Time: 10: 37)

$$(x_b - x_a) \frac{\partial u}{\partial t} = -\{ \psi(u(x_b) - u(x_a)) \}$$

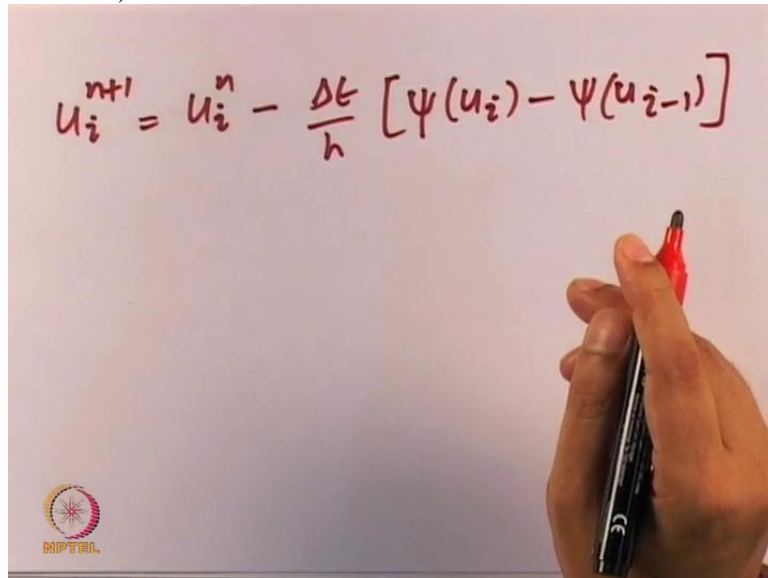
$\Delta x = h \quad u^* \approx u$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -\frac{1}{h} [\psi(x_{i+1/2}) - \psi(x_{i-1/2})]$$

$$u_i^{n+1} = u_i^n + \left(-\frac{\Delta t}{h}\right) [\psi(x_{i+1/2}) - \psi(x_{i-1/2})]$$

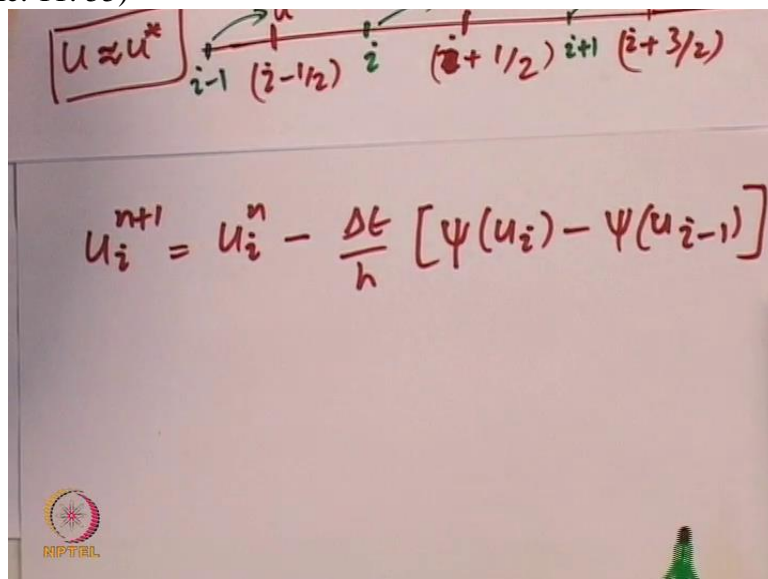
We saw during the Maxwell equation problems that we can go higher order in approximating the flux functions on the surface using either muscle method of the method of waiting so this is a very simple straight forward approach which works fine for simple problem like heat conduction problems where dissipation is not an issue so we can still use such an approximation.

(Refer Slide Time: 11: 06)


$$u_i^{n+1} = u_i^n - \frac{\Delta t}{h} [\psi(u_i) - \psi(u_{i-1})]$$

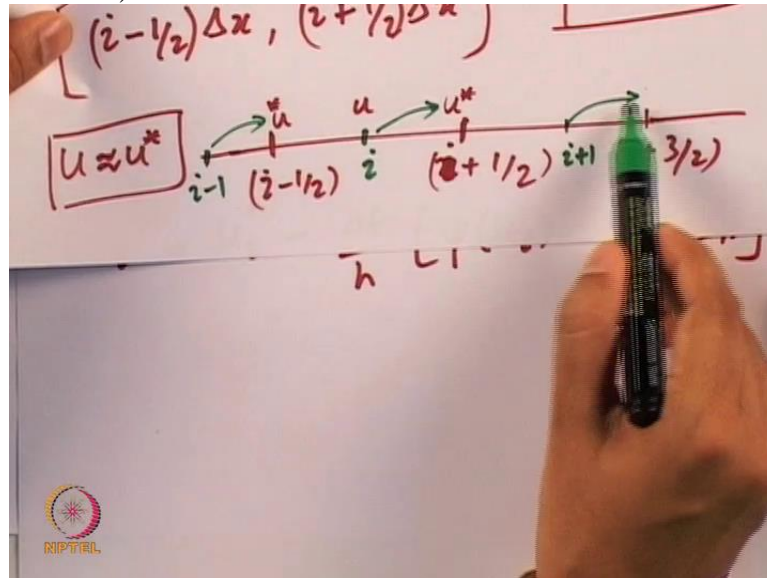
So what we get essentially is U of i n plus 1 is equal to u of i n so i will take the minus sign which was here outside and write it directly using a minus sign. So I will have minus delta t divided by h . I will simplify it saying $[\psi(u_i) - \psi(u_{i-1})]$ How did I come to this point will be clear if I once again show you the stencil here.

(Refer Slide Time: 11: 55)


$$u_i^{n+1} = u_i^n - \frac{\Delta t}{h} [\psi(u_i) - \psi(u_{i-1})]$$

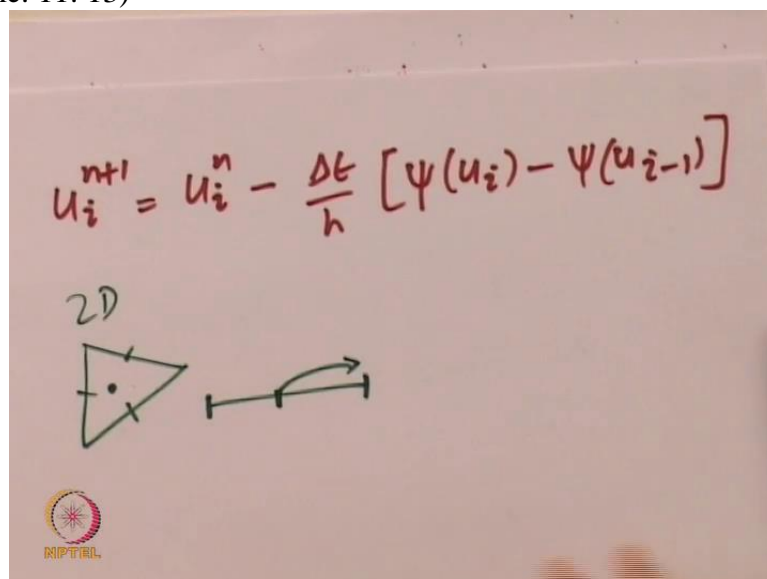
So the flux at i plus half is going to be directly given by the value of u itself u at i so that is what I have written here u_i . Whereas for this u this i minus half the value at this point will be given by the cell that is sitting here. We have not written it but this is going to be the i minus 1 so that will give me the value of the field here this will give me the value of the field here. Similarly this will give the value of the field here so on and so forth.

(Refer Slide Time: 12: 38)



So this is the way we are computing the value. So i minus 1 will contribute to i minus half i will contribute to i plus half i plus 1 will contribute to i plus 3 by 2. So the value at the face centres here the faces are nothing but the end of the each of the elements in one dimension.

(Refer Slide Time: 11: 13)



In three dimensions you might talk about the face centres using the triangles. So the value at this face centre the value at this face centre the value at this face centre so this will be the 2D case. But in the 1D case we are talking about face centres at the end of the edges. So the value at this point is going to be the value at this point. And the value of the neighbouring cell before will contribute to the value here so on and so forth. That is why we have i minus 1 here.

(Refer Slide Time: 13: 36)

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{h} [\psi(u_i) - \psi(u_{i-1})]$$

2D

$$\psi(u_i) = c u_i$$

$$u_i^{(n+1)} = u_i^{(n)} - \frac{c \Delta t}{h} (u_i^{(n)} - u_{i-1}^{(n)})$$

Explicit Formulation.

And if we can simplify the subscripts a little bit and write the value of the flux, so flux of (u_i) is equal to c multiplied by u of i where c is the propagation velocity of the solution in Maxwell equation the c will be the velocity of light itself the velocity of electromagnetic field that is propagating whereas in this problem more general c will be the velocity of the solution propagation. So u_i is propagation. So we use this notation we can simplify this equation even further as u_i^{n+1} is equal to u_i^n minus $c \Delta t$ divided by h multiplied by $(u_i^n - u_{i-1}^n)$. So what is beautiful about this equation is the terms what we had on left hand side are using $n+1$. Whereas the terms on the right hand side are having values only at u^n so this is an explicit formulation.

(Refer Slide Time: 15: 15)

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{h} [\psi(u_i) - \psi(u_{i-1})]$$

2D

$$\psi(u_i) = c u_i$$

$$0 < \frac{c \Delta t}{h} \leq 1$$

CFL stability

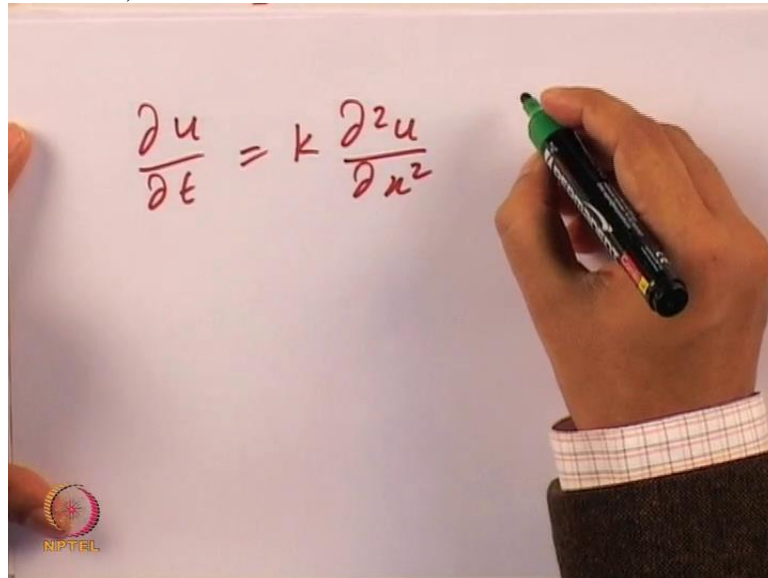
$$u_i^{(n+1)} = u_i^{(n)} - \frac{c \Delta t}{h} (u_i^{(n)} - u_{i-1}^{(n)})$$

Explicit Formulation.

And of course what we have also is there is a condition of CFL which says $0 < C \Delta t / h \leq 1$ so this is a CFL like condition this is for stability and this is

a value for plus function and this is a final formulation that we have with this background we are going to study a simple problem using finite element method the problem what we're going to do is a heat conduction problem and the formulation of the equation of the problem is going to look like this

(Refer Slide Time: 16: 20)

A hand holding a green marker is writing the partial differential equation $\frac{\partial u}{\partial t} = k \frac{\partial^2 u}{\partial x^2}$ on a whiteboard. The equation is written in red ink. In the bottom left corner of the whiteboard, there is a small circular logo with the text "NPTEL" below it.
$$\frac{\partial u}{\partial t} = k \frac{\partial^2 u}{\partial x^2}$$

So we will have $\frac{\partial u}{\partial t}$ is equal to k some constant multiplied by $\frac{\partial^2 u}{\partial x^2}$. So this is a problem where we are going to use finite volume method and we are going to compare it also with the the finite difference method that we studied before because there is a lot of similarity at least on the one dimensional problem you will see the finite difference and finite volume will have some similarities and some differences.

(Refer Slide Time: 16: 57)

$$\frac{\partial u}{\partial t} = k \frac{\partial^2 u}{\partial x^2} \quad u(x,0) = \cancel{0.1} = 0.1 \sin(4\pi x/L)$$

$L = \text{length of domain } [0, 1] \quad L = 1$

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

So let's take this particular problem and the problem has certain boundary conditions and some initial conditions the initial condition is $u(x,0)$ for all x at time equal to 0, the solution is going to have the following relationship. So $u(x)$ is equal to $0.1 \sin 4\pi x$ by L ; L is the length of the domain. So we go from 0 to 1, so L will be equal to 1.

And what is also important to know is there is a boundary condition which is nothing but $u(0,t)$ is going to be 1 and $u(L,t)$ that is going to be equal to 0. So L is equal to 1 so we can write $u(1,t)$ is equal to 0.

(Refer Slide Time: 18: 20)

$$\frac{\partial u}{\partial t} = k \frac{\partial^2 u}{\partial x^2} \quad u(x,0) = \cancel{0.1} = 0.1 \sin(4\pi x/L)$$

$L = \text{length of domain } [0, 1] \quad L = 1$

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

So if you look at this problem what you will have as some initial condition certain patterns the pattern is going to have $4\pi x$ by L as the value so if you look at it and try to draw it so this is X equal to let's say 1 and X equal to zero; At X equal to zero a solution is going to have value that is equal to 1 so if we say this is one search going to have a value 1 but it is

also going to have some reputation because it's a sinusoidal function and it is going to have a value of zero at the end and the in between 0 and 1 it is going to go through certain parameters so if you put X equal to zero then this what you will get is the value 0 but it will also have a value that is one because it is being said as 1 so this is the boundary condition this is a natural condition in which it propagates so let's see how we can simulate this problem using finite volume. So for that we have to start with discretising the left and right hand side what you will do is we will do Central differences both for the time and for the special derivative.

(Refer Slide Time: 19: 57)

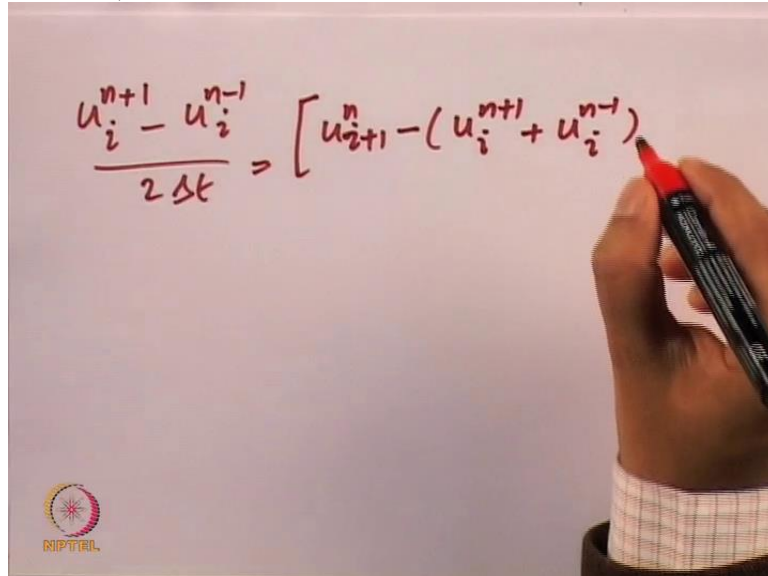
$$\frac{\partial u}{\partial t} = \frac{u_i^{n+1} - u_i^{n-1}}{2 \Delta t} \quad \begin{array}{l} i: \text{spatial ordinates} \\ n: \text{temporal ordinates} \end{array}$$

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2}$$

$$u_i^n = \frac{u_i^{n+1} + u_i^{n-1}}{2} \Rightarrow \boxed{2u_i^n = u_i^{n+1} + u_i^{n-1}}$$

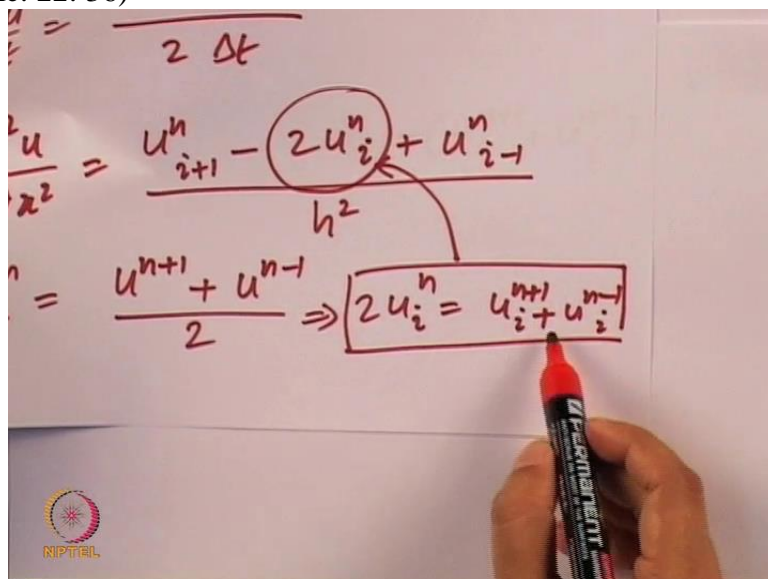
So what we have got for $\frac{\partial u}{\partial t}$ by $\frac{\partial u}{\partial x}$ using the central difference in will be $u(n+1) - u(n-1)$ divided by $2 \Delta t$ this we know from our finite difference method modules and when we want to do the same thing for the spatial derivative but with the second order this is going to be given by $u_{i+1} - 2u_i + u_{i-1}$ divided by h^2 . This is central differencing space and what you are going to do is we are going to substitute for this one using averaging so what you are going to do is basically we are going to say u_i^n is going to be given by $\frac{u_i^{n+1} + u_i^{n-1}}{2}$. So I am taking the old value which is $u(n-1)$ and the current value is $u(n)$ future value is going to be $u(n+1)$ so I am taking the old value plus the new value divided by 2 this will give me $2u_i^n = u_i^{n+1} + u_i^{n-1}$.

(Refer Slide Time: 21: 50)


$$\frac{u_i^{n+1} - u_i^{n-1}}{2 \Delta t} = [u_{i+1}^n - (u_i^{n+1} + u_i^{n-1})]$$

So this is what I am going to substitute into this equation which will get transformed into a different form which is nothing but u_{i+1}^n so I am substituting the value for the left hand side as well u_i^{n-1} divided by $2 \Delta t$ that is equal to $[u_{i+1}^n - (u_i^{n+1} + u_i^{n-1})]$

(Refer Slide Time: 22: 36)


$$\frac{2u}{\Delta x^2} = \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2}$$
$$u_i^n = \frac{u_i^{n+1} + u_i^{n-1}}{2} \Rightarrow \boxed{2u_i^n = u_i^{n+1} + u_i^{n-1}}$$

This is what I am substituting here so this should have i , these are the subscripts. So once I substitute that here what will happen is I will have this two terms here

(Refer Slide Time: 22: 45)

$$\frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t} = \frac{[u_{i+1}^n - (u_i^{n+1} + u_i^{n-1}) + u_{i-1}^n]}{h^2}$$

$$u_i^{n+1} = \frac{1-\beta}{1+\beta} u_i^{n-1} + \frac{\beta}{1+\beta} [u_{i+1}^n + u_{i-1}^n]$$

where $\beta = \frac{2k\Delta t}{h^2}$

And then I will have the last term unchanged [u i plus 1 n minus (u n plus 1 i plus u n minus 1 i) plus u n i minus 1] divided by h square. And if I rearrange it I keep only u n plus 1 i so there is another term of u n plus 1,i here so when I bring them back all in one side what I will get is u n plus 1 i equal to 1 minus Beta divided by 1 plus Beta of u i n minus 1 plus Beta divided by 1 plus Beta [u i plus 1 n plus u n i minus 1]; where beta equal to 2k delta t divided by h square.

(Refer Slide Time: 24: 02)

$$\frac{\partial u}{\partial t} = k \frac{\partial^2 u}{\partial x^2}$$

$$u(x,0) = 0.1 \sin(4\pi x/L)$$

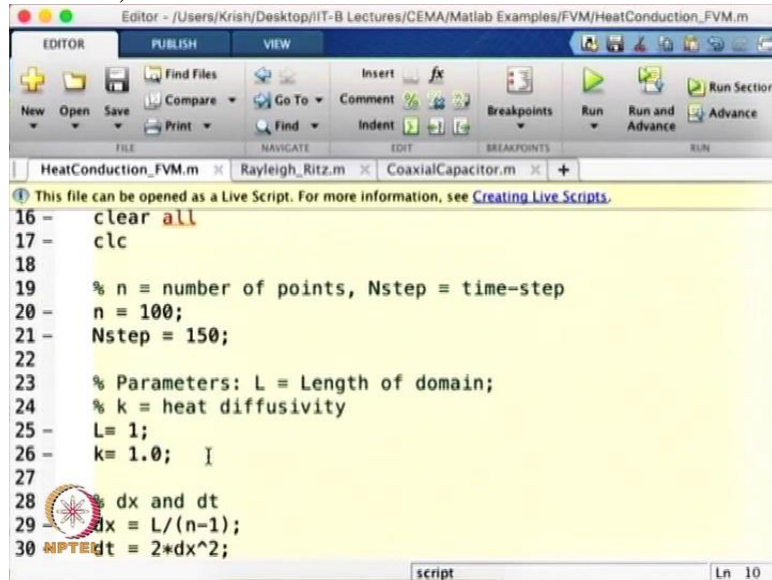
$L = \text{length of domain } [0,1] \quad L=1$

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

$$u(L,t) = 0$$

k is the constant that we had in the starting of the equation and this is the way the Beta is calculated. So there is a lot of similarity between this approach and finite difference approach and of course these are some differences. The differences are nothing but the way we are computing the fluxes and how we are approximating the fluxes but now let us look into the program itself and see how the initial value and final value are being replicated

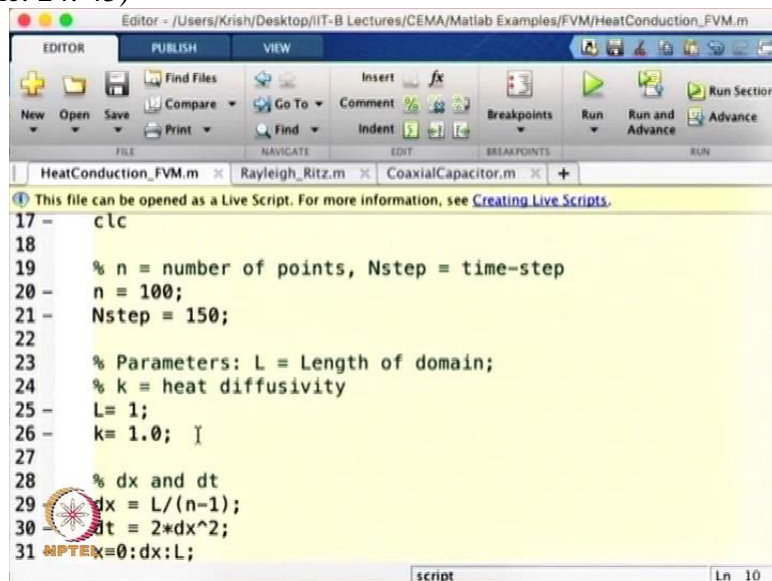
(Refer Slide Time: 24: 36)



```
Editor - /Users/Krish/Desktop/IIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment Insert Breakpoints Run Run and Advance
Find Print Indent Find Breakpoints Run and Advance
FILE NAVIGATE EDIT BREAKPOINTS RUN
HeatConduction_FVM.m x Rayleigh_Ritz.m x CoaxialCapacitor.m x +
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
16 - clear all
17 - clc
18
19 % n = number of points, Nstep = time-step
20 - n = 100;
21 - Nstep = 150;
22
23 % Parameters: L = Length of domain;
24 % k = heat diffusivity
25 - L = 1;
26 - k = 1.0; I
27
28 % dx and dt
29 dx = L/(n-1);
30 NPTER dt = 2*dx^2;
script Ln 10
```

So this is the program so I am going to write it for 150 steps. The parameters are L equal to the domain length

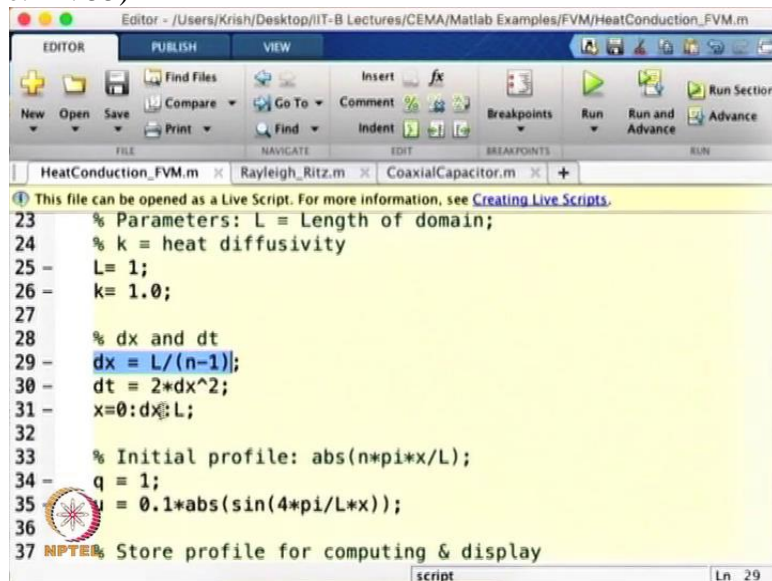
(Refer Slide Time: 24: 45)



```
Editor - /Users/Krish/Desktop/IIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment Insert Breakpoints Run Run and Advance
Find Print Indent Find Breakpoints Run and Advance
FILE NAVIGATE EDIT BREAKPOINTS RUN
HeatConduction_FVM.m x Rayleigh_Ritz.m x CoaxialCapacitor.m x +
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
17 - clc
18
19 % n = number of points, Nstep = time-step
20 - n = 100;
21 - Nstep = 150;
22
23 % Parameters: L = Length of domain;
24 % k = heat diffusivity
25 - L = 1;
26 - k = 1.0; I
27
28 % dx and dt
29 dx = L/(n-1);
30 NPTER dt = 2*dx^2;
31 NPTER x=0:dx:L;
script Ln 10
```

K is the heat diffusivity; I said L equal to 1 and K equal to 1.

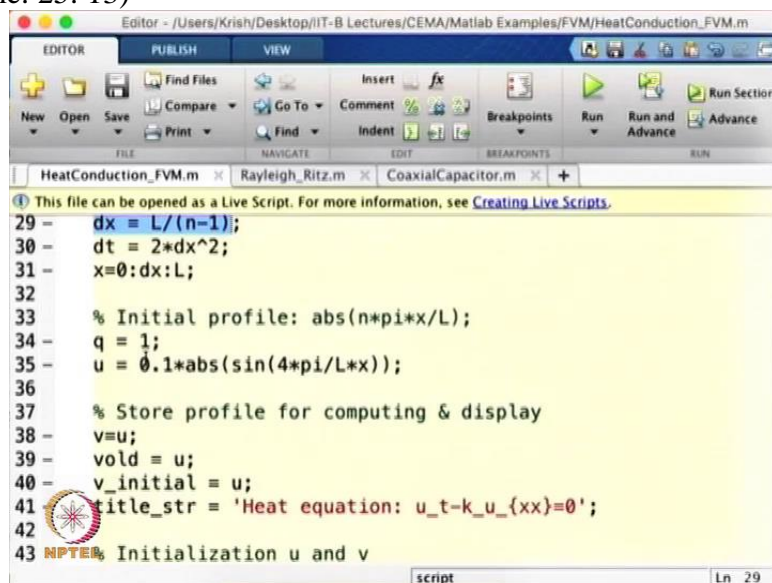
(Refer Slide Time: 24: 55)



```
Editor - /Users/Krish/Desktop/IIIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment Insert Breakpoints Run Run and Advance
Find Indent Breakpoints Run Run and Advance
FILE NAVIGATE EDIT BREAKPOINTS RUN
HeatConduction_FVM.m x Rayleigh_Ritz.m x CoaxialCapacitor.m x +
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
23 % Parameters: L = Length of domain;
24 % k = heat diffusivity
25 - L = 1;
26 - k = 1.0;
27
28 % dx and dt
29 - dx = L/(n-1);
30 - dt = 2*dx^2;
31 - x=0:dx:L;
32
33 % Initial profile: abs(n*pi*x/L);
34 - q = 1;
35 - u = 0.1*abs(sin(4*pi/L*x));
36
37 NPTEB Store profile for computing & display
script Ln 29
```

And the other parameters dx equal to L divided by (n minus 1). So this is going to be the h value as well. And dt is going to be dt is equal to twice the dx square value

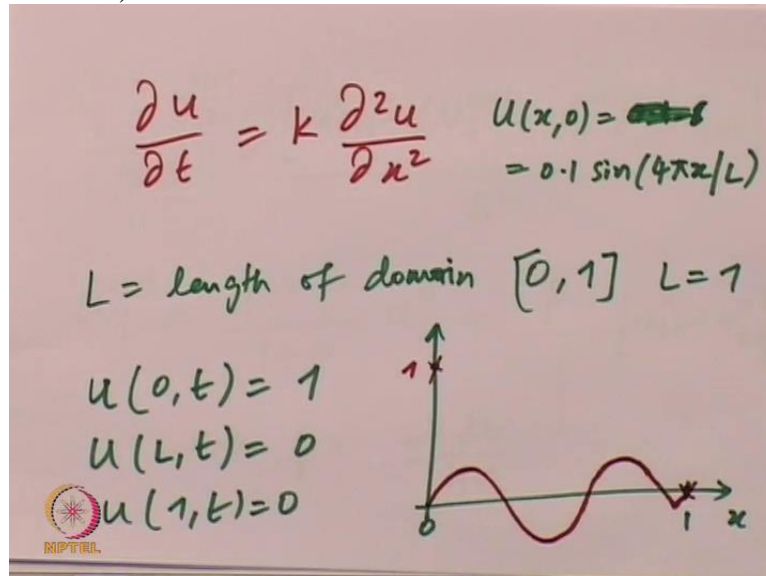
(Refer Slide Time: 25: 13)



```
Editor - /Users/Krish/Desktop/IIIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment Insert Breakpoints Run Run and Advance
Find Indent Breakpoints Run Run and Advance
FILE NAVIGATE EDIT BREAKPOINTS RUN
HeatConduction_FVM.m x Rayleigh_Ritz.m x CoaxialCapacitor.m x +
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
29 - dx = L/(n-1);
30 - dt = 2*dx^2;
31 - x=0:dx:L;
32
33 % Initial profile: abs(n*pi*x/L);
34 - q = 1;
35 - u = 0.1*abs(sin(4*pi/L*x));
36
37 % Store profile for computing & display
38 - v=u;
39 - vold = u;
40 - v_initial = u;
41 - title_str = 'Heat equation: u_t-k_u_{xx}=0';
42
43 NPTEB Initialization u and v
script Ln 29
```

And we are setting the initial profile q equal to 1.

(Refer Slide Time: 25: 20)



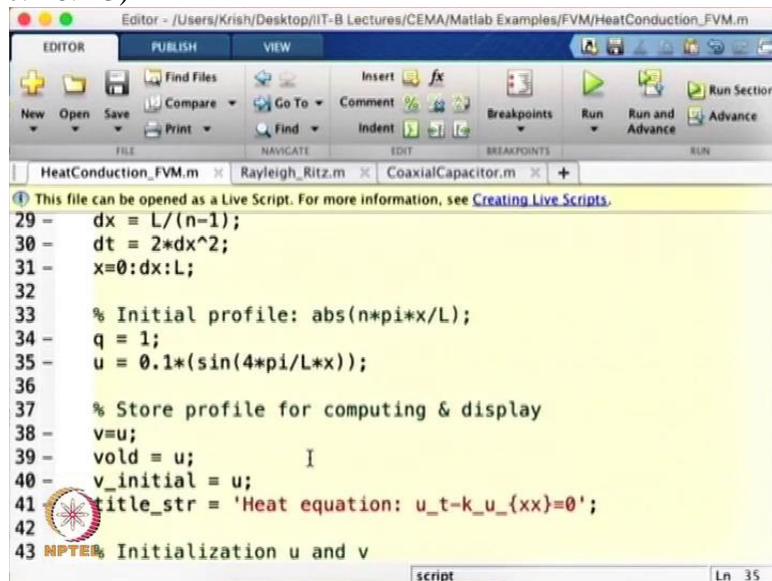
So I said the value is going to be 1 for x equal to 0. And 0 for x equal to 1 so these two points are fixed and in between it is going to have a sinusoidal variation and L we have set it to 1. So it is going to be 4 pi of x. So it will have some kind of at this point it is going to be equal to 0. So if we are simulating this problem and we are going to see what is going to happen.

(Refer Slide Time: 26: 00)

```
Editor - /Users/Krish/Desktop/IIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
New Open Save Compare Find Files Go To Comment Insert fx Breakpoints Run Run and Advance Advance
FILE NAVIGATE EDIT BREAKPOINTS RUN
HeatConduction_FVM.m x Rayleigh_Ritz.m x CoaxialCapacitor.m x +
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
25 - L = 1;
26 - k = 1.0;
27
28 % dx and dt
29 - dx = L/(n-1);
30 - dt = 2*dx^2;
31 - x=0:dx:L;
32
33 % Initial profile: abs(n*pi*x/L);
34 - q = 1;
35 - u = 0.1*(sin(4*pi/L*x));
36
37 Store profile for computing & display
38 v=u;
39 NPTEN vold = u;
```

So go back to the code what we see is we are setting the values for the initial profile

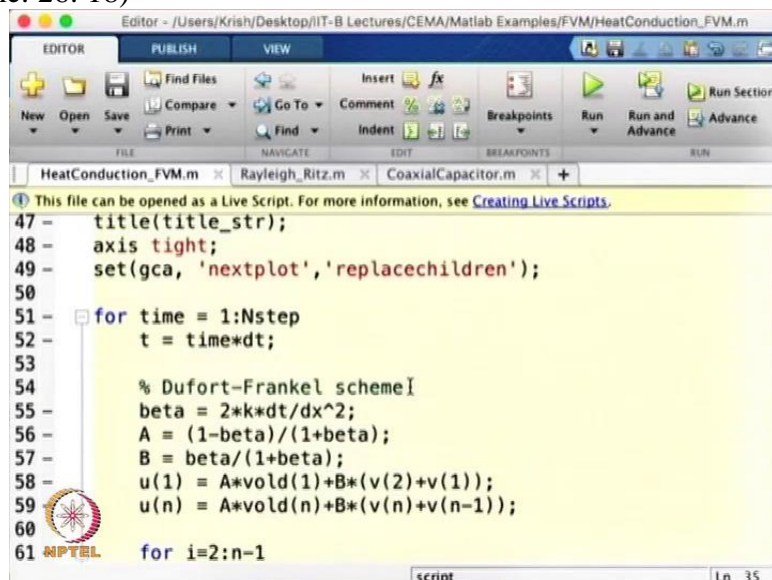
(Refer Slide Time: 26: 13)



```
Editor - /Users/Krish/Desktop/IIIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment Insert Breakpoints Run Run and Advance
HeatConduction_FVM.m Rayleigh_Ritz.m CoaxialCapacitor.m
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
29 - dx = L/(n-1);
30 - dt = 2*dx^2;
31 - x=0:dx:L;
32
33 % Initial profile: abs(n*pi*x/L);
34 - q = 1;
35 - u = 0.1*(sin(4*pi/L*x));
36
37 % Store profile for computing & display
38 - v=u;
39 - vold = u;
40 - v_initial = u;
41 - title_str = 'Heat equation: u_t-k_u_{xx}=0';
42
43 NPTEL Initialization u and v
```

And this is going to be the value that we are giving

(Refer Slide Time: 26: 18)



```
Editor - /Users/Krish/Desktop/IIIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment Insert Breakpoints Run Run and Advance
HeatConduction_FVM.m Rayleigh_Ritz.m CoaxialCapacitor.m
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
47 - title(title_str);
48 - axis tight;
49 - set(gca, 'nextplot', 'replacechildren');
50
51 for time = 1:Nstep
52 - t = time*dt;
53
54 % Dufort-Frankel scheme
55 - beta = 2*k*dt/dx^2;
56 - A = (1-beta)/(1+beta);
57 - B = beta/(1+beta);
58 - u(1) = A*vold(1)+B*(v(2)+v(1));
59 - u(n) = A*vold(n)+B*(v(n)+v(n-1));
60
61 NPTEL for i=2:n-1
```

And we are trying to plot the initial value and also the computed value the computed value we are using the method called as Dufort Frankel Scheme. And Dufort Frankel Scheme is nothing but the method that we have used here.

(Refer Slide Time: 26: 35)

$$\frac{\partial u}{\partial t} = \frac{u^{n+1} - u^{n-1}}{2 \Delta t}$$
$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2}$$
$$u_i^n = \frac{u_i^{n+1} + u_i^{n-1}}{2} \Rightarrow \boxed{2u_i^n = u_i^{n+1} + u_i^{n-1}}$$

Dufort-Frankel Scheme

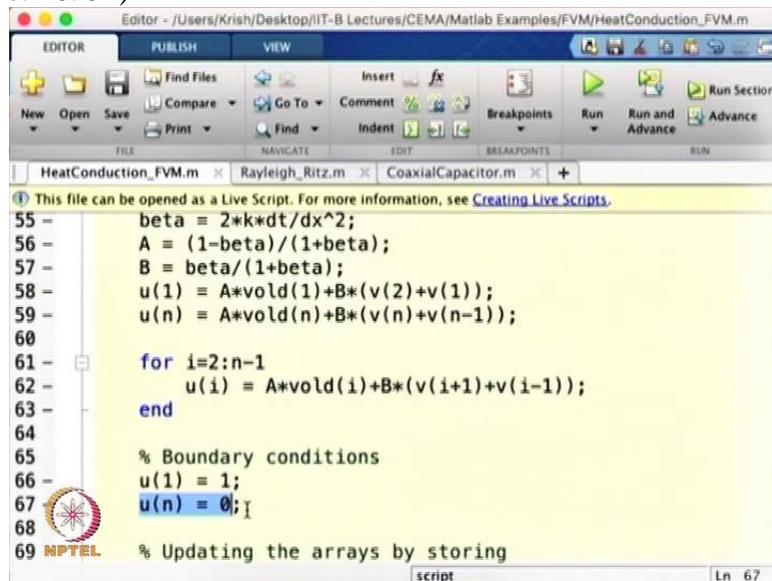
So this is for u^n Dufort Frankel Scheme; substituting for u_i^n we are substituting the value that is average of the 2 time step values

(Refer Slide Time: 26: 48)

```
Editor - /Users/Krish/Desktop/IT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m
EDITOR PUBLISH VIEW
+ Find Files Insert fx Breakpoints Run Run and Advance
New Open Save Compare Go To Comment % Indent Find Breakpoints Run Run and Advance
FILE NAVIGATE EDIT BREAKPOINTS RUN
HeatConduction_FVM.m Rayleigh_Ritz.m CoaxialCapacitor.m +
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
50
51 - for time = 1:Nstep
52 -     t = time*dt;
53
54     % Dufort-Frankel scheme
55 -     beta = 2*k*dt/dx^2;
56 -     A = (1-beta)/(1+beta);
57 -     B = beta/(1+beta);
58 -     u(1) = A*vold(1)+B*(v(2)+v(1));
59 -     u(n) = A*vold(n)+B*(v(n)+v(n-1));
60
61 -     for i=2:n-1
62 -         u(i) = A*vold(i)+B*(v(i+1)+v(i-1));
63 -     end
64
```

So that is what we have done here in the code.

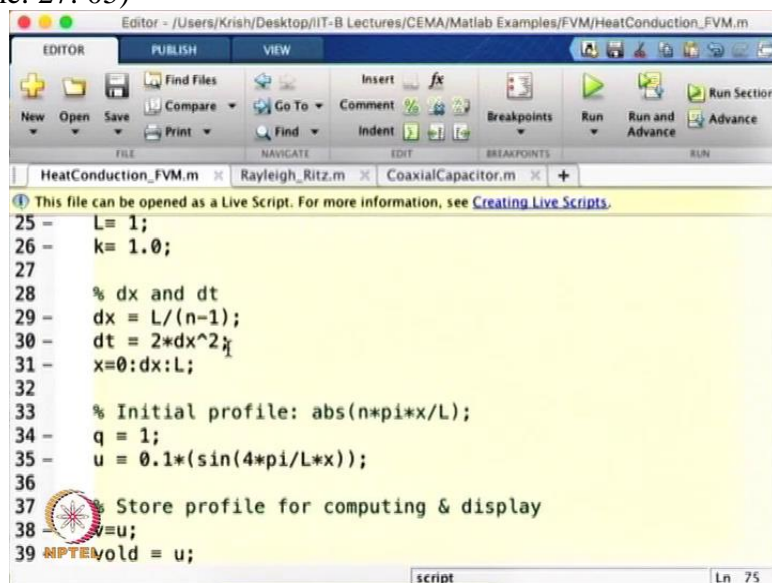
(Refer Slide Time: 26: 52)



```
HeatConduction_FVM.m
55 - beta = 2*k*dt/dx^2;
56 - A = (1-beta)/(1+beta);
57 - B = beta/(1+beta);
58 - u(1) = A*vold(1)+B*(v(2)+v(1));
59 - u(n) = A*vold(n)+B*(v(n)+v(n-1));
60
61 - for i=2:n-1
62 -     u(i) = A*vold(i)+B*(v(i+1)+v(i-1));
63 - end
64
65 - % Boundary conditions
66 - u(1) = 1;
67 - u(n) = 0;
68
69 - % Updating the arrays by storing
```

And we have set the boundary condition to be equal to 1 boundary condition at $u(1)$ which is the last point is going to be 0.

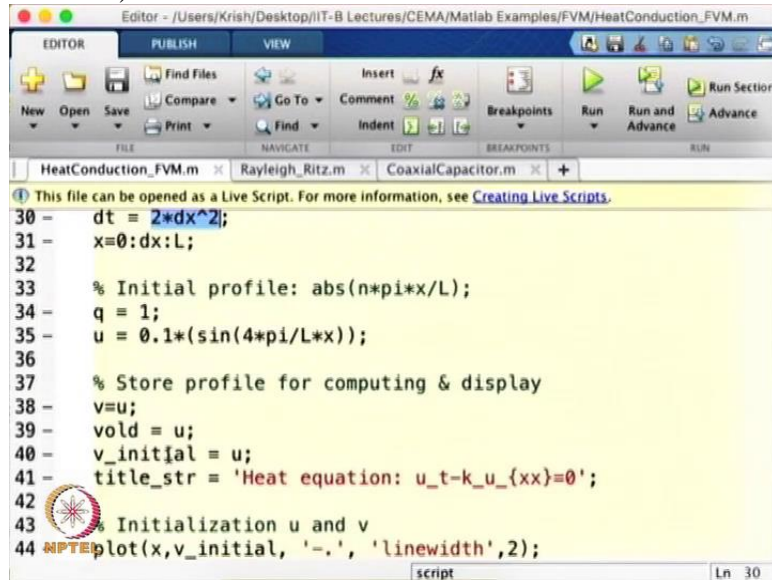
(Refer Slide Time: 27: 05)



```
HeatConduction_FVM.m
25 - L = 1;
26 - k = 1.0;
27
28 - % dx and dt
29 - dx = L/(n-1);
30 - dt = 2*dx^2;
31 - x=0:dx:L;
32
33 - % Initial profile: abs(n*pi*x/L);
34 - q = 1;
35 - u = 0.1*(sin(4*pi/L*x));
36
37 - Store profile for computing & display
38 - v=u;
39 - vold = u;
```

And the value of the updated equation is also going to get plotted so initially we will have v initial which is going to be given by the dotted lines. And then we are going to plot the updated equation so what we will see is the wave heat conduction is going to move. And we are going to plot the value of the u at time step t equal to 0.03 which is nothing but

(Refer Slide Time: 27: 40)

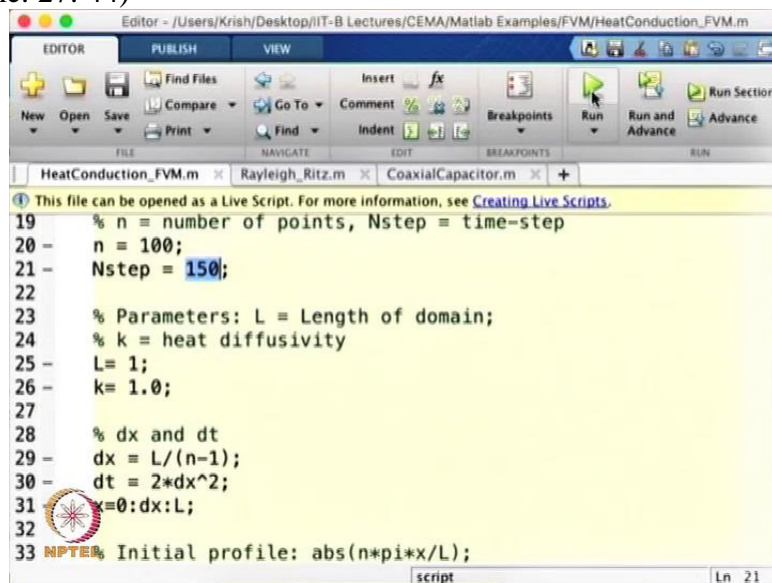


The image shows a MATLAB editor window with the following code:

```
HeatConduction_FVM.m
30 - dt = 2*dx^2;
31 - x=0:dx:L;
32
33 % Initial profile: abs(n*pi*x/L);
34 - q = 1;
35 - u = 0.1*(sin(4*pi/L*x));
36
37 % Store profile for computing & display
38 - v=u;
39 - vold = u;
40 - v_initial = u;
41 - title_str = 'Heat equation: u_t-k_u_{xx}=0';
42
43 % Initialization u and v
44 NPTEP plot(x,v_initial, '-.', 'linewidth',2);
```

This delta t multiplied by the number of iteration we are running

(Refer Slide Time: 27: 44)

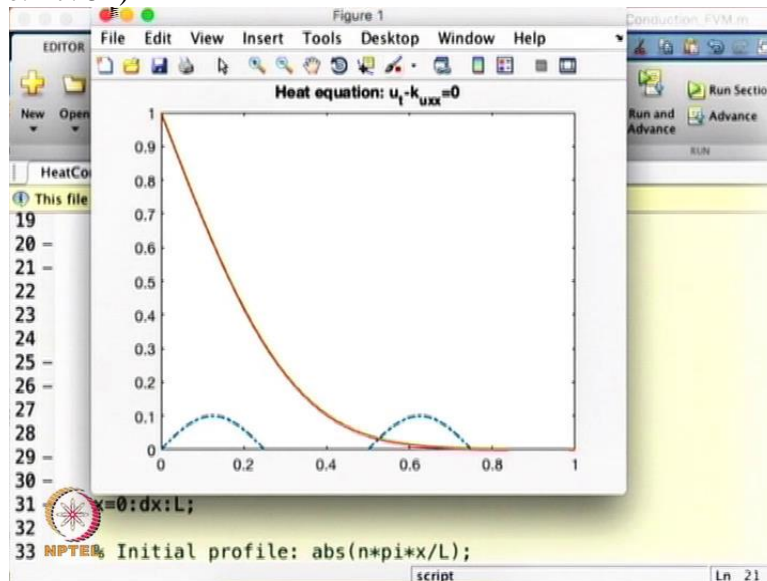


The image shows a MATLAB editor window with the following code:

```
HeatConduction_FVM.m
19 % n = number of points, Nstep = time-step
20 - n = 100;
21 - Nstep = 150;
22
23 % Parameters: L = Length of domain;
24 % k = heat diffusivity
25 - L= 1;
26 - k= 1.0;
27
28 % dx and dt
29 - dx = L/(n-1);
30 - dt = 2*dx^2;
31 - x=0:dx:L;
32
33 NPTEP % Initial profile: abs(n*pi*x/L);
```

So number of iteration we are running is 150 so if we do 150 multiplied by delta t we will get that. So let us run it and see what is happening.

(Refer Slide Time: 27: 54)



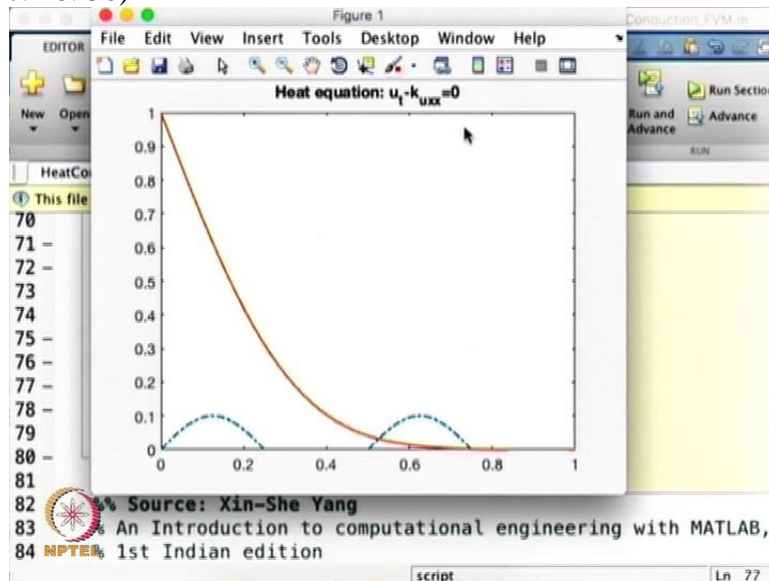
So this is a dashed line you see that the value is 1 initially and then it goes to 0. And you see that the value of the computed solution after the time step is 150 steps forward. We say that it is propagating in this manner. So if we can re run again we will see.

(Refer Slide Time: 28: 20)

```
Editor - /Users/Krish/Desktop/IIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m*
EDITOR PUBLISH VIEW
New Open Save Compare Go To Comment % Indent Breakpoints Run Run and Advance
HeatConduction_FVM.m* x Rayleigh_Ritz.m x CoaxialCapacitor.m x +
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
70 % results into v
71 vold= v;
72 v = u;
73
74 %Plotting the results
75 plot(x,v_initial, '-.',x,u,'linewidth',2);
76 axis([0 L 0 1]);
77 pause(0.001);
78 drawnow;
79
80 end
81
82 % Source: Xin-She Yang
83 % An Introduction to computational engineering with MATLAB,
84 NPTEL, 1st Indian edition
script Ln 77
```

I will have to rerun it with little bit more time pause. So let me put time pause is equal to 0.001

(Refer Slide Time: 28: 30)



I will reduce the time stepping further

(Refer Slide Time: 28: 40)

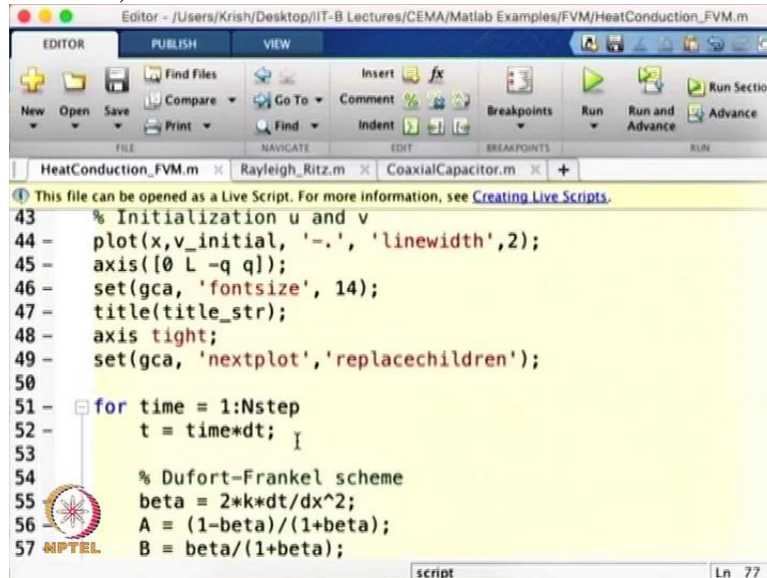
Editor - /Users/Krish/Desktop/IIIT-B Lectures/CEMA/Matlab Examples/FVM/HeatConduction_FVM.m*

70 % results into v
71 vold= v;
72 v = u;
73
74 %Plotting the results
75 plot(x,v_initial, '-.',x,u,'linewidth',2);
76 axis([0 L 0 1]);
77 pause(0.0001);
78 drawnow;
79
80 end
81
82 % Source: Xin-She Yang
83 % An Introduction to computational engineering with MATLAB,
84 NPTEL, 1st Indian edition

script Ln 77

I am reducing the pausing between the simulation even more.

(Refer Slide Time: 28: 45)



```
43 % Initialization u and v
44 - plot(x,v_initial, '-.', 'linewidth',2);
45 - axis([0 L -q q]);
46 - set(gca, 'fontsize', 14);
47 - title(title_str);
48 - axis tight;
49 - set(gca, 'nextplot','replacechildren');
50
51 - for time = 1:Nstep
52 -     t = time*dt;
53
54     % Dufort-Frankel scheme
55     beta = 2*k*dt/dx^2;
56     A = (1-beta)/(1+beta);
57     B = beta/(1+beta);
```

So this is the way the heat conduction is going to propagate and the initial values are given by the dotted lines and then the final value is given by the solid line at time step 150. So this is the very good example for you to try and the source of the code from this book An Introduction to Computational Engineering with Matlab. I encourage you to practice coding in One dimension and two dimension which will be very good for you to understand the methods, mathematics and also the complication. It is not good to go directly to three dimensional problem and code it. So we have tested finite volume method both using a one dimensional problem like a simple heat conduction problem. And I also shown you earlier some of the problems that we simulated using the finite volume method like the horn antenna or the spiral antenna or the wave guide truncation and the PML applications.

So in that way we have covered quite a bit of simulation using finite volume method so I encourage you to program your own code at least for one dimensional problems so that you get to know the method more and more closely and you can differentiate various aspects of the method compared to finite element method or finite difference method that we can master the technique and also know when the method can be applied and when it cannot Thank You!