NPTEL
NATIONAL PROGRAMME ON
TECHNOLOGY ENHANCED LEARNING

IIT BOMBAY

CDEEP
IIT BOMBAY

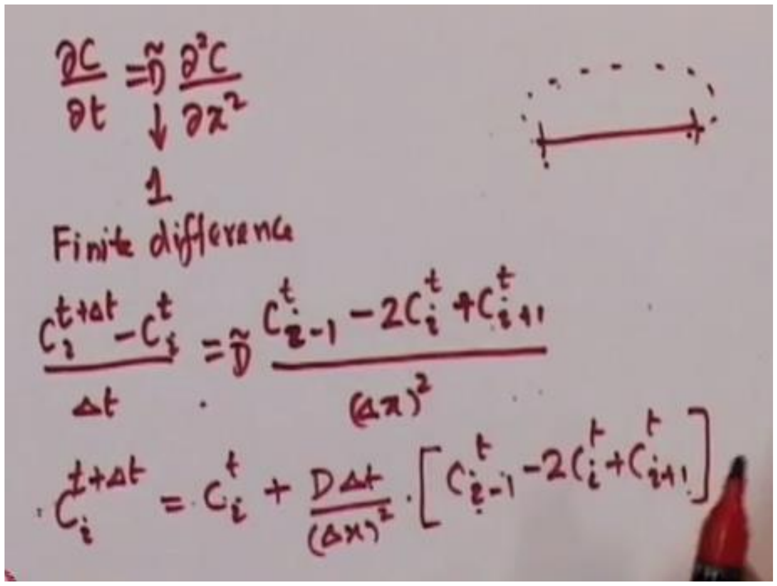
Phase field modeling; the materials science, mathematics and computational aspects

Prof. M P Gururajan
Department of Metallurgical Engineering
And materials Science, IIT Bombay

Module No.15
Lecture No.62
Numerical solution of CH:
Finite difference method

Welcome we want to solve numerically let us start with the diffusion equation we have probably solved it earlier too but for the sake of continuity and for the sake of comparison with the faithful model it is better to do it once more.

(Refer Slide Time: 00:33)



$$\frac{\partial C}{\partial t} = \tilde{D} \frac{\partial^2 C}{\partial x^2}$$

$$\tilde{D} \downarrow 1$$

Finite difference

$$\frac{C_i^{t+\Delta t} - C_i^t}{\Delta t} = \tilde{D} \frac{C_{i-1}^t - 2C_i^t + C_{i+1}^t}{(\Delta x)^2}$$

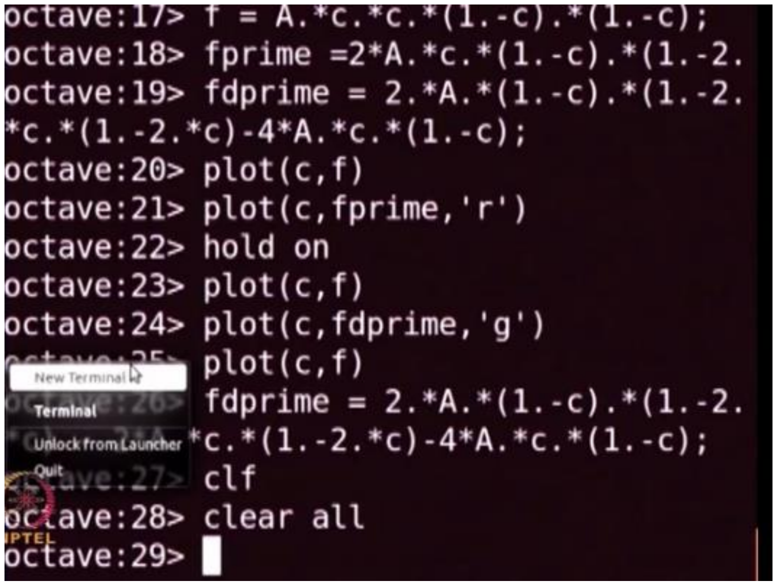
$$C_i^{t+\Delta t} = C_i^t + \frac{D\Delta t}{(\Delta x)^2} \cdot \left[C_{i-1}^t - 2C_i^t + C_{i+1}^t \right]$$

So we have the equation $\partial C/ \partial t = \partial^2 C/ \partial x^2$ some D~ which I am always going to assume to be 1 because this TX everything is non-dimensional, okay so I am going to do a finite difference scheme so and so what happens so we have $C^{t+ \Delta T} - C^t / \Delta t$ is equal to so diffusivity is one and I want to do this so this is nothing but $C_{i-1} - 2C_i + C_{i+1} / \Delta x^2$ okay so let me keep the detail it is one.

But anyway we will keep it for this purpose so I am going to do it explicit so I am going to take all this to be time t and so then I know that C at time T + ΔT is nothing but C at the position so this is all I at any I position is time T+ D $\Delta t/ \Delta x^2$ multiplying the $C_{i-1}{}^t$ -2 Ci^t + $C_{i+1}{}^t$ and like we said earlier I am going to use periodic boundary condition because when I have the domain like this for this point there is no I – 1 point.

So I am going to use that as the I - 1 point for this point there is no I + 1 point I am going to use this as the I + 1 point so in some sense it is like this, so we are going to use the cyclic or periodic boundary condition, okay so I want to right now a script which will do this and solve the diffusion equation. And the initial profile that now I am going to assume is going to be a sinusoidal profile so let us take a very small system so that I can run it in a short amount of time and look at what happens to these solutions okay.

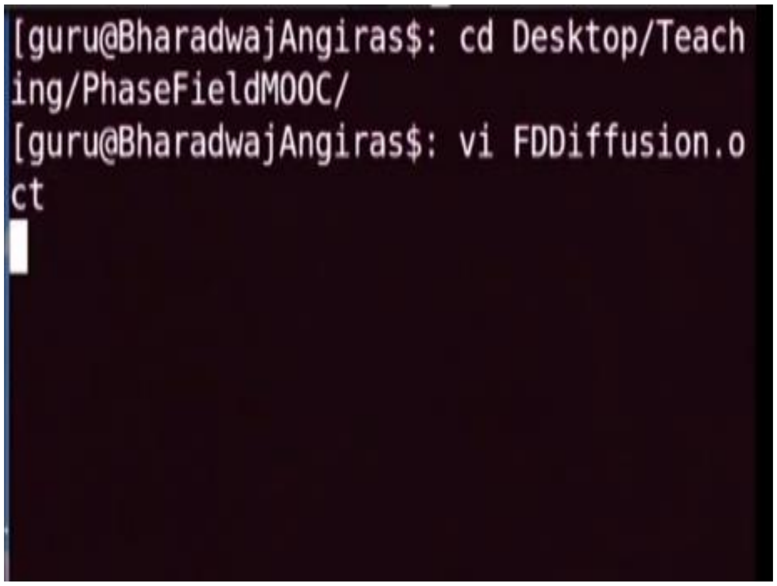
(Refer Slide Time: 02:41)



```
octave:17> f = A.*c.*c.*(1.-c).*(1.-c);
octave:18> fprime =2*A.*c.*(1.-c).*(1.-2.
octave:19> fdprime = 2.*A.*(1.-c).*(1.-2.
*c.*(1.-2.*c)-4*A.*c.*(1.-c);
octave:20> plot(c,f)
octave:21> plot(c,fprime,'r')
octave:22> hold on
octave:23> plot(c,f)
octave:24> plot(c,fdprime,'g')
octave:25> plot(c,f)
octave:26> fdprime = 2.*A.*(1.-c).*(1.-2.
*c.*(1.-2.*c)-4*A.*c.*(1.-c);
octave:27> clf
octave:28> clear all
octave:29> 
```

So let me take a solution and this time we are going to write it using scripts, okay.

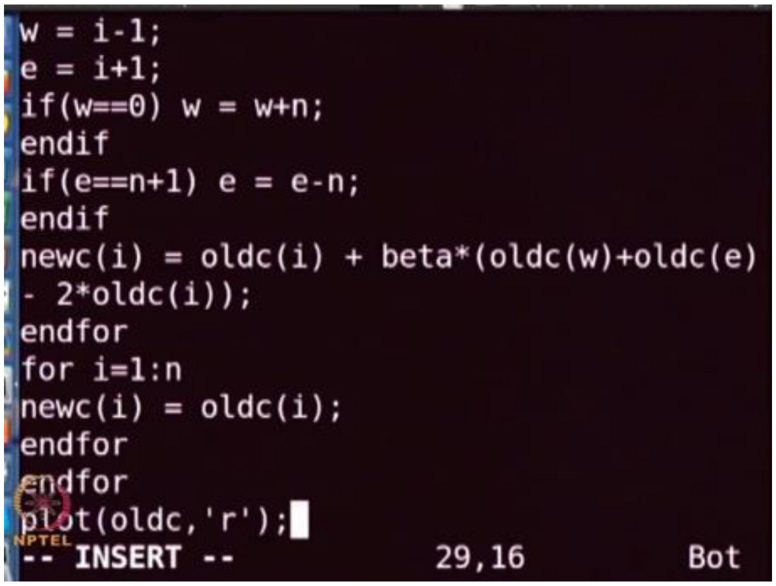
(Refer Slide Time: 02:470



```
[guru@BharadwajAngiras$: cd Desktop/Teach
ing/PhaseFieldMOOC/
[guru@BharadwajAngiras$: vi FDDiffusion.o
ct
```

So I have to go to the directory in which okay so I am going to make finite difference at diffusion .oct so that is the file okay.

(Refer Slide Time: 03:05)



```
w = i-1;
e = i+1;
if(w==0) w = w+n;
endif
if(e==n+1) e = e-n;
endif
newc(i) = oldc(i) + beta*(oldc(w)+oldc(e)
- 2*oldc(i));
endfor
for i=1:n
newc(i) = oldc(i);
endfor
endfor
plot(oldc,'r');
-- INSERT --              29,16          Bot
```

So I am going to generate a script so I am going to take some BT, so delta T is 0.001 let us say and then I have h = 1 so let me call it as dx okay so it is easier for us to follow dx is 1 and I am going to call β as this d times and let me define D as one, so d $\Delta t/ \Delta x^2$ is β, okay. So that is d*dt / (dx * dx) so that is my β okay so I am going to take a system with the 32 points okay and the total number of time steps that I am going to run the simulation is 40,000, okay.

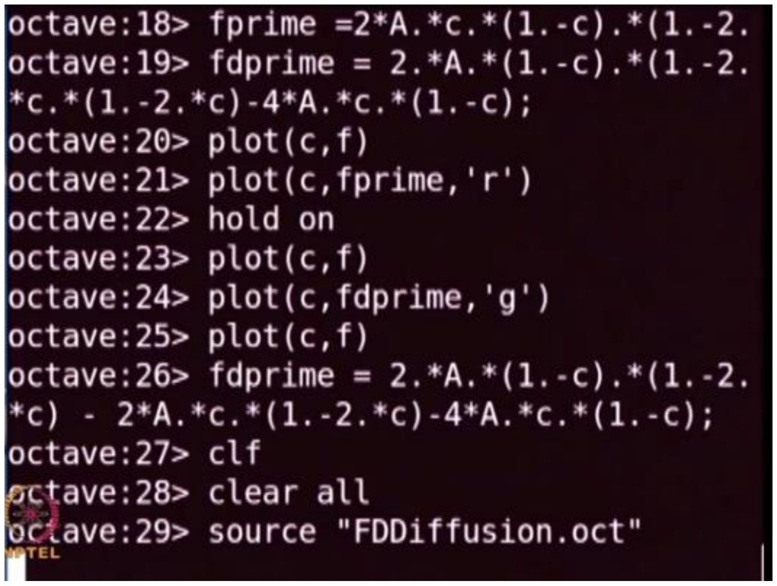
So I am going to remove any old figure so that is clear figure I am going to use hold on so that I can plot the initial profile as well as the later profile so I am going to take m = 1 so m basically tells how many wavelengths we are going to incorporate in the given domain so this is initial profile that I am trying to make so far I = 1 to N okay, what do I do? Old c(i) is nothing but 0.5 * 1 + Sin okay.

What is the sine wave that I take, 2 * pi * m * I * dx/ N okay, so that is a new C(i) let me now start with 0 end for, okay so I have initialized the new C to be 0 and old C is a sinusoidal profile and how many wavelengths are there is decided by this M, so one wavelength is what we are starting the simulation bit, so let me plot the initial profile okay, so this is for plotting the initial profile then I say for j = 1: N this is the time loop and what we are going to do for I = 1 to N for every point what are we going to do the west point is equal to I - 1 is point is equal to i + 1right now we have to implement periodic condition so boundary condition.

So if w is 0 that means we are at the first point so for that the west point is the last point end if and if east point is equal to n + 1 that means you are at the last point and then the east point is equal to east minus M right, so that is because the first point is the east point for the last point now so ne c(i) is nothing but old c(i) + ß & (old c(w) + old c(e) – 2 * old c(i) itself, right. So this is what we have and we end the loop for the all the points.

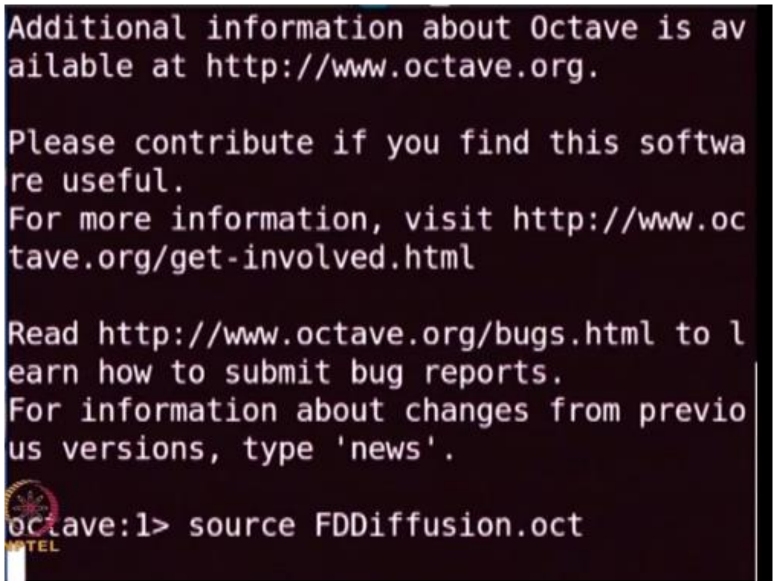
And so now we update the profile so we say for i = 1 :N and new c(I) is basically old c(i) end for right and then I am going to end the time loop, so the this is this end far is basically for this loop that I am running the time steps that I am running. So that is the end for so now I can plot the final profile so I am going to plot the old C that I have now and I am going to plot it at the red line. So red is basically the final profile so that we know so this is a script.

(Refer Slide Time: 07:52)


```
octave:18> fprime =2*A.*c.*(1.-c).*(1.-2.
octave:19> fdprime = 2.*A.*(1.-c).*(1.-2.
*c.*(1.-2.*c)-4*A.*c.*(1.-c);
octave:20> plot(c,f)
octave:21> plot(c,fprime,'r')
octave:22> hold on
octave:23> plot(c,f)
octave:24> plot(c,fdprime,'g')
octave:25> plot(c,f)
octave:26> fdprime = 2.*A.*(1.-c).*(1.-2.
*c) - 2*A.*c.*(1.-2.*c)-4*A.*c.*(1.-c);
octave:27> clf
octave:28> clear all
octave:29> source "FDDiffusion.oct"
```

So I am going to go to octave and I am going to source FD diffusion .doc okay. So this is now taking a sinusoidal profile and it is going to give me what happens at the end so we have taken one sinusoidal profile and the red line is going to say what happens after some time to this profile so that is what we are looking for in this case okay, so it is going to take some time.

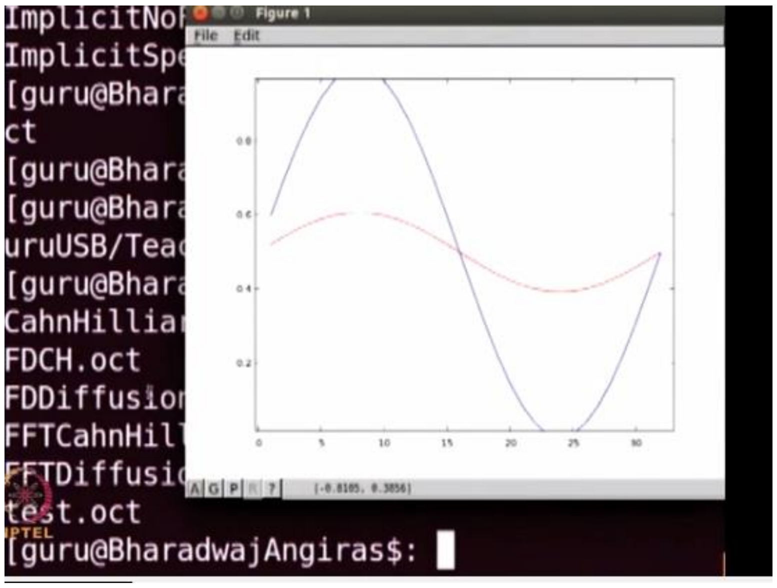
(Refer Slide Time: 08:30)



```
Additional information about Octave is av
ailable at http://www.octave.org.

Please contribute if you find this softwa
re useful.
For more information, visit http://www.oc
tave.org/get-involved.html

Read http://www.octave.org/bugs.html to l
earn how to submit bug reports.
For information about changes from previo
us versions, type 'news'.

octave:1> source FDDiffusion.oct
```

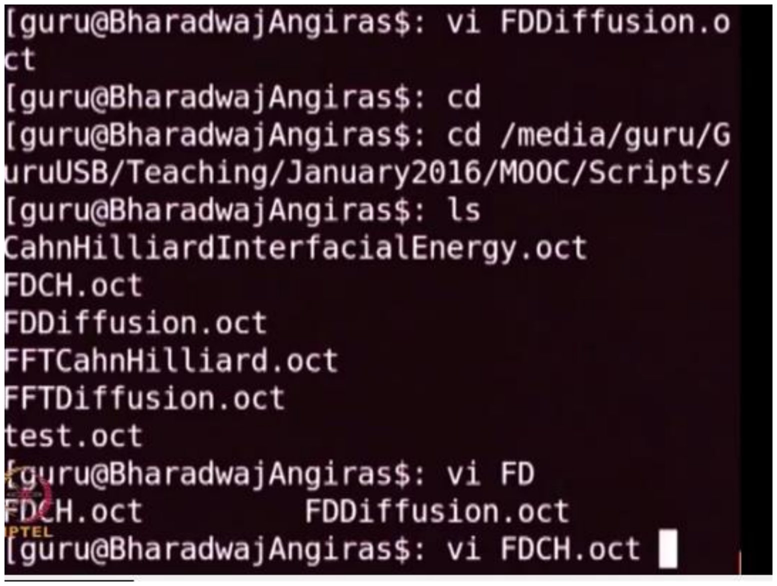
Even though it is a very small system you know it's not big it is only 32 points but it can take a longtime especially because we are taking40,000 times sighs so I think i forgot to put hold on so let me because I am just finding the final solution but that does not make sense unless we know what the initial profile looks like there is a hold on. So let me I am going to put the hold on here okay. So let me rerun again, so I am going to plot old C here with a blue line by default that is what it should the plot octave source diffusivity okay.

(Refer Slide Time: 11:46)


```
ImplicitNo
ImplicitSp
[guru@Bhar
ct
[guru@Bhar
[guru@Bhar
uruUSB/Tea
[guru@Bhar
CahnHillia
FDCH.oct
FDDiffusio
FFTCahnHil
FFTDiffusi
test.oct
[guru@BharadwajAngiras$:
```

It is not clear what is going wrong, so let me go to file that I so yes so here we have the initial profile which is plotted in blue and the red profile is basically the profile after forty thousand times steps because each time step is 0,001 it is 40,000 x 0.001 so that is near about 40 time units in non-dimensional time manners, so as you can see because this is a purely diffusion equation whatever be the wavelength. That it is trying to go to the homogeneous solution so it will reach a 0.5 everywhere so that is the solution that it is trying to get to.

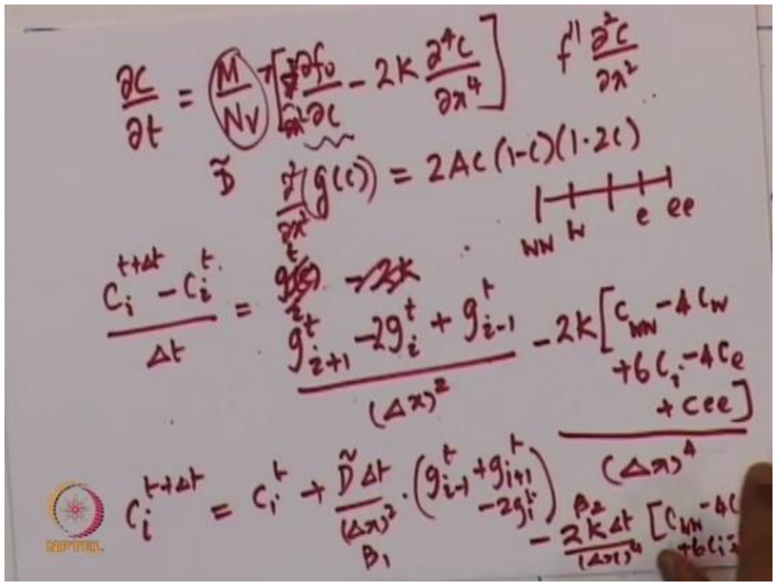
(Refer Slide Time: 14:39)



```
[guru@BharadwajAngiras$: vi FDDiffusion.o
ct
[guru@BharadwajAngiras$: cd
[guru@BharadwajAngiras$: cd /media/guru/G
uruUSB/Teaching/January2016/MOOC/Scripts/
[guru@BharadwajAngiras$: ls
CahnHilliardInterfacialEnergy.oct
FDCH.oct
FDDiffusion.oct
FFTCahnHilliard.oct
FFTDiffusion.oct
test.oct
[guru@BharadwajAngiras$: vi FD
FDCH.oct          FDDiffusion.oct
[guru@BharadwajAngiras$: vi FDCH.oct
```

Now we can take her the same kind of finite difference code but with Cornelia now, so how do we do that so let us write down the Cornelia equation.

(Refer Slide Time: 14:59)



$$\frac{\partial c}{\partial t} = \frac{M}{N_v}\left[\frac{\partial f_0}{\partial c} - 2K\frac{\partial^4 c}{\partial x^4}\right] \qquad f'' \frac{\partial^2 c}{\partial x^2}$$

$$\tilde{D} \qquad \frac{\partial^2}{\partial x^2} g(c) = 2Ac(1-c)(1-2c)$$

$$\frac{c_i^{t+\Delta t} - c_i^t}{\Delta t} = \frac{g_{i+1}^t - 2g_i^t + g_{i-1}^t}{(\Delta x)^2} - 2K\frac{\left[c_{ww} - 4c_w + 6c_i - 4c_e + c_{ee}\right]}{(\Delta x)^4}$$

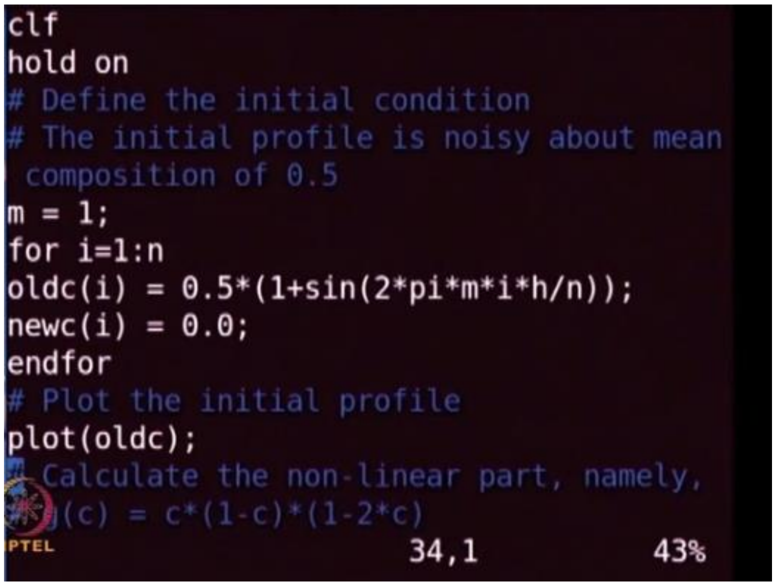
$$c_i^{t+\Delta t} = c_i^t + \underbrace{\frac{\tilde{D}\Delta t}{(\Delta x)^2}}_{B_1}\cdot\left(g_{i-1}^t + g_{i+1}^t - 2g_i^t\right) - \overset{B_2}{\frac{2K\Delta t}{(\Delta x)^4}}\left[c_{ww} - 4c \ldots + 6c_i \ldots\right]$$

So Cornelia equation is $\partial C/\partial t$ = M/Nv into so I am going to go back to the form in which we wrote first So $\partial F_0/\partial c - 2K\, \partial^4 C/\partial x^4$ okay, now this quantity $\partial F_0/\partial c$ I am going to call as some g(c) and g (c) is nothing but 2AC (1- c) (1 – 2c) right so now we want to do so we are going to use a similar technique like we used last time for the finite difference scheme, so it is $C_I^{T+\Delta t} - C_i^t / \Delta t$ equal to. So this is some non dimensional one that I am going to assume so this g (c) now I have to take g (c) okay and this g (c) will be taken at time T for every position I okay.

Necessarily – 2K now this is the fourth derivative for which I have to take the finite difference scheme there is a Δ^2 so there is a Δ^2 acting on this so okay so the Δ^2 so this then becomes so we have to take the second derivative 4g because $\Delta^2\, \partial_0/\partial tC$ is what we wrote as $\partial^2 C/\partial x^2$ but now that is by ∂x^2 acting on c $\partial^2/\partial x^2$ acting on g (c) so that is going to be g at position at time T at position I + 1 – g_i^t plus this is two times g_i^t to $g_{i-1}^t / \Delta x^2$ right so this is the first term and -2K this is the fourth derivative term so for fourth derivative term then we have to go so if you have a point and if you have a West Point and west of West Point.

And you have to have a east point and the east of east point then in this case it is written as $C_{WW} - 4\, C_W + 6$ C at the ith position -4 $C_e + C_{EE}$ okay so that is what this is and this is 8 and -8 so that so this is the discretization and in this case it has to be divided by power ΔX^4 okay, so now you can write this expression so $C_i^{t} + \Delta t = C_I^{t}$ plus some $\beta\ \Delta t / \Delta x^2$ so let us call this some D~ $\Delta t / \Delta x^2$ with this quantity $g_{i-1}^{t} + g_{i+1}^{t} - 2g_i^{t}$ so that is one part -2K $\Delta t/(\Delta x^2)^4$ right. kΔt, Δx^4 so this is another quantity so if you call this as β1 so this is some β2 multiply by this quantity C_{WW} and minus $4C_W + 6C_i - 4C_E + C_{EE}$ right so that is that is the discretization and this is the finite-difference discretization and this is what we want to implement now on the computer so here is the code.

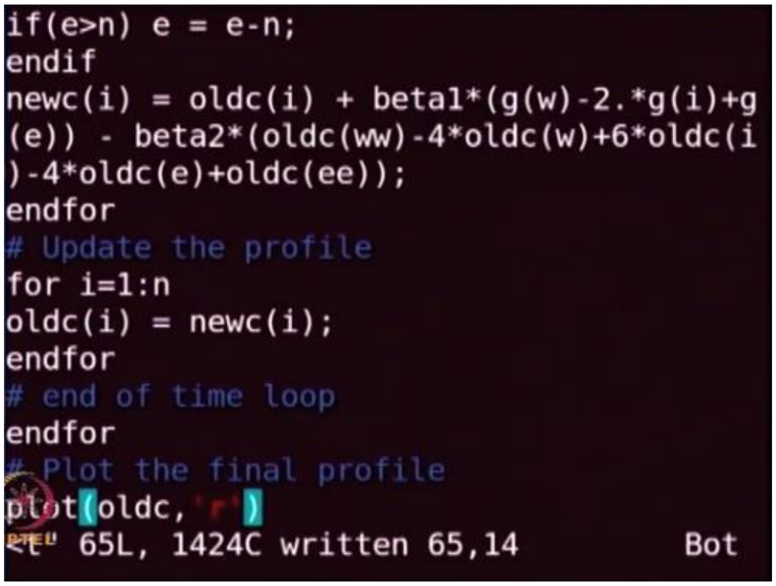
(Refer Slide Time: 19:27)



```
clf
hold on
# Define the initial condition
# The initial profile is noisy about mean
 composition of 0.5
m = 1;
for i=1:n
oldc(i) = 0.5*(1+sin(2*pi*m*i*h/n));
newc(i) = 0.0;
endfor
# Plot the initial profile
plot(oldc);
# Calculate the non-linear part, namely,
 g(c) = c*(1-c)*(1-2*c)
                                34,1          43%
```

So that we first we define Δt and we define a soul let me call this as dx and β1is basically dt/ (dx * dx) and β2 is dx/dx because kappa okay, so let me put that also d = 1. 0 and Kappa = 1.0 so this basically d * dt and so this should be 2 * K * dt/ dx * dx * dx okay, so to make things simpler so I again take a 32 and 40, 000 time test so a clear figure I hold on and again I take one wavelength. And so I have the old C so I have put a sinusoidal profile new C is 0 end for at what the initial profile.

(Refer Slide Time: 20:39)

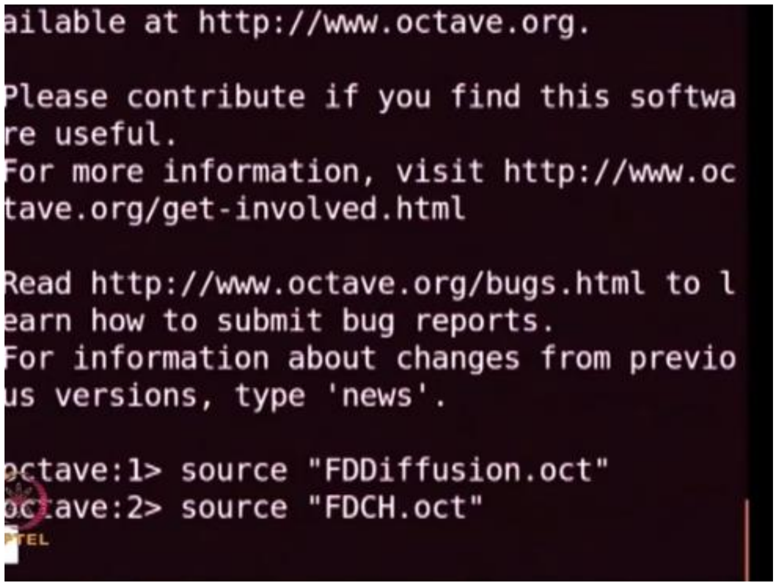


```
if(e>n) e = e-n;
endif
newc(i) = oldc(i) + beta1*(g(w)-2.*g(i)+g
(e)) - beta2*(oldc(ww)-4*oldc(w)+6*oldc(i
)-4*oldc(e)+oldc(ee));
endfor
# Update the profile
for i=1:n
oldc(i) = newc(i);
endfor
# end of time loop
endfor
# Plot the final profile
plot(oldc, 'r')
"" 65L, 1424C written 65,14          Bot
```

So that will be plotted with blue by default and now I have the for i = 1 to N g(i) is 2 old Ci – 1- old C $_I$ – 1 – old C$_i$ so this is how we calculate g, g was 2, A is 1 so let me define that also A = 1so this is 2 * A * C * (1-c) * (1 – 2c) So we now start the time stepping we again go w is I-1 ww Is I-2, E is i +1, e is i+2 so if W becomes less than 1 then that you make it plus n and similarly e becomes greater than n you make it e-1.

And for a w also, so new C(i) is old C(i) β * g(w) – 2g(i) + g(e) – β2 * (old c (ww) – 4 old c(w) and so on and so forth, okay. So this is the now the profile has to be updated so the this is the new C we got so this becomes a old C now in the next step and then we keep going for it, okay. So this is the code now if we execute this part.

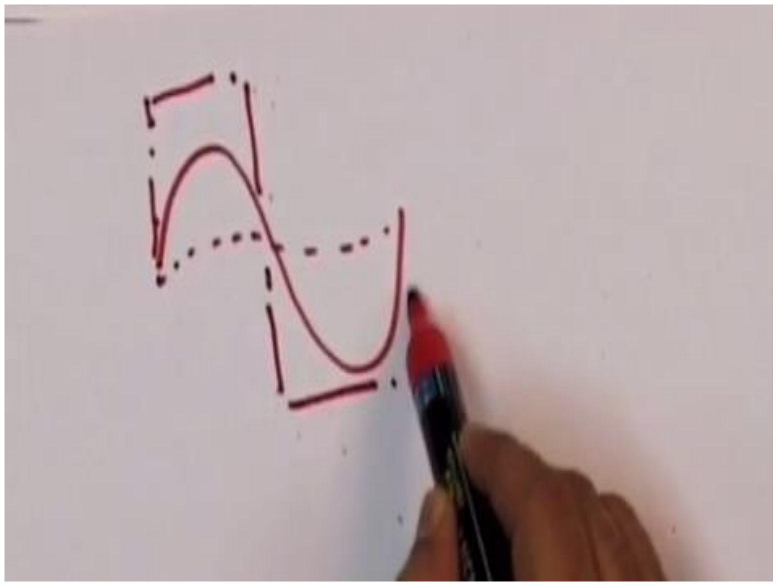
(Refer Slide Time: 22:13)



```
ailable at http://www.octave.org.

Please contribute if you find this softwa
re useful.
For more information, visit http://www.oc
tave.org/get-involved.html

Read http://www.octave.org/bugs.html to l
earn how to submit bug reports.
For information about changes from previo
us versions, type 'news'.

octave:1> source "FDDiffusion.oct"
octave:2> source "FDCH.oct"
```

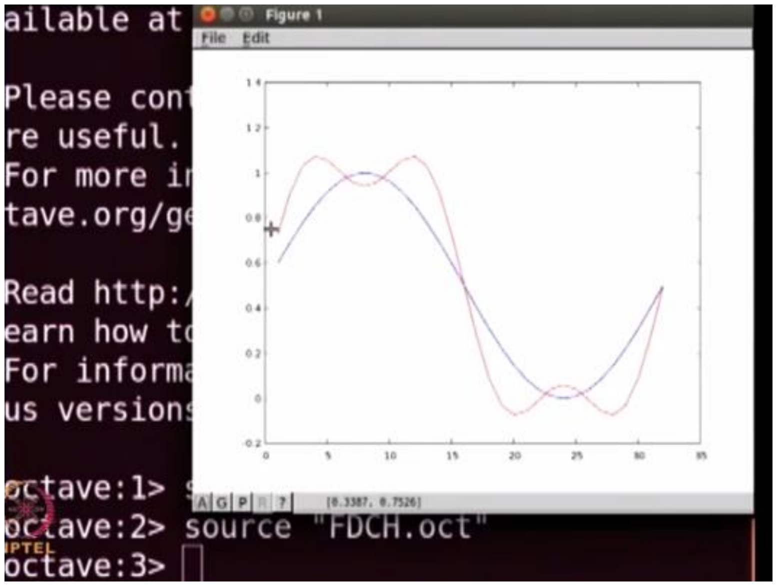
So we free source of FD -Conn Hilliard .oct what do we get, you will see that in this case we have put a sinusoidal wave and we know that in the case of Con Hilliard equation the solution should become if you have a sinusoidal wave then that part which is having a curvature like this should become a profile which will go to one and the lower portion that should go to 0.

(Refer Slide Time: 22:52)



So in if you if you take a profile like this so if you take a profile like this and in the case of diffusion equation you expect a profile to become like this in the case of corn Hilliard you expect the profile to become like this, right, the profile has come out.

(Refer Slide Time: 23:11)



```
ailable at

Please con
re useful.
For more i
tave.org/g

Read http:/
earn how t
For inform
us version

octave:1> s
octave:2> source "FDCH.oct"
octave:3> 
```

So you can see that the initial profile that we started with is the sinusoidal profile in the case of diffusion it became a curve which was going towards homogeneous 0.5 line but in the case of corn Hilliard it is becoming a curve like this what I would expect is that it becomes a 1 here and then it forms an interface and becomes 0 here and that is another interface because of periodic boundary condition.

However you see that this equilibration has not happened it has this kind of structure because it is going to take some time before you know the profile help becomes the correct profile that is the problem with finite difference tactic because we are restricted to taking a Δ T of 0.001 okay and that itself is very small and so the method is A there are accuracy problems minor difference technique is not very accurate that the error goes as especially because we are taking the fourth derivative.

So there are a higher amount of error that you accumulate because of this discretization on top of it because it is an explicit technique the time step that you can take is itself very small and that does not allow me to actually evolve the system for longer duration and get the proper profile so what this is why we use the Fourier transform technique okay so in the next part of this lecture I want to show how the diffusion equation and the continuity equation are solved using the Fourier transform technique. And there we will see that we will be able to take much larger time step so we can evolve it fast and we can actually go to the equilibrium profile so that is what I would want to do in the next part of this lecture, thank you.