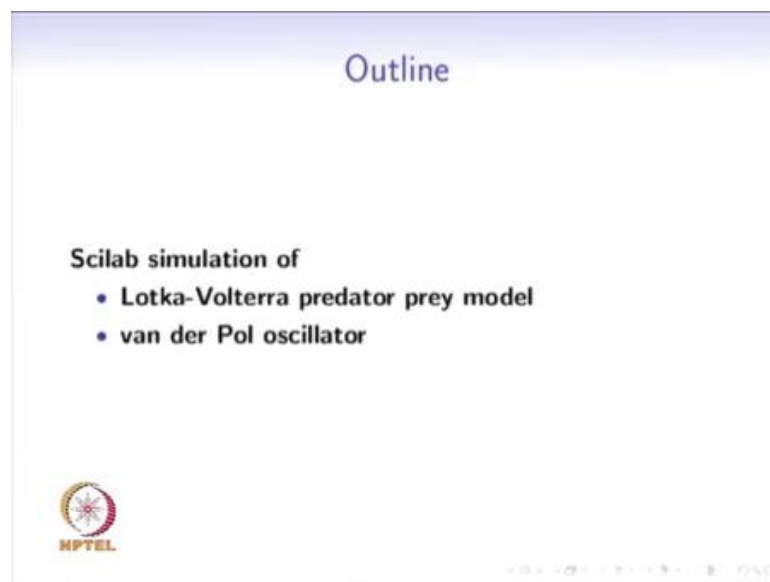


Nonlinear Dynamical Systems
Prof. Madhu. N. Belur and Prof. Harish. K. Pillai
Department of Electrical Engineering
Indian Institute of Technology, Bombay

Lecture - 12
Scilab simulation of – Lotka – Volterra Predator Prey Model
Van Der Pol Oscillator
Review of Linearization
Operating Point / Operating Trajectory

Welcome to lecture number 12 on non-linear dynamical systems. So, we have seen a detailed analysis of the Lotka Volterra predator prey model and about the Van Der Pol oscillator.

(Refer Slide Time: 00:32)



Today we will see how we can visualize the dynamics of these two important non-linear systems using a simulation done in a package called scilab. So, scilab is free and open source and it is helpful to use scilab for understanding, how the dynamics of various systems governed by differential equations can be simulated. So, let us just briefly see the Lotka Volterra predator prey model.


(Refer Slide Time: 00:57)

Lotka Volterra Predator Prey Model

Predator → Hunter
Population dynamics of two species: prey and hunter.
Equations of the model

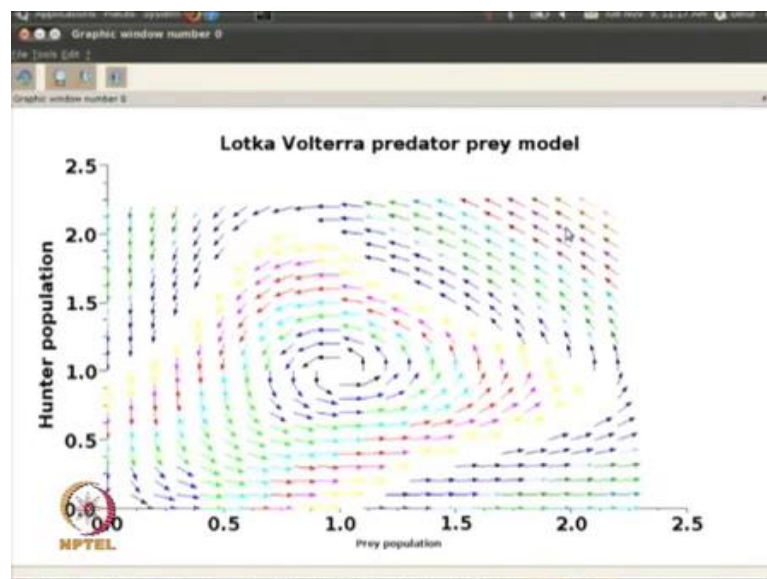
$$\begin{aligned}\dot{x}_h &= -x_h + x_h x_p \\ \dot{x}_p &= x_p - x_h x_p\end{aligned}$$

x_h is the amount of hunter specimen in the model
 x_p is the amount of prey specimen in the model



So, even though the word predator is commonly used, we will use hunter to denote the predator specie. So, we have two species here: one is the prey, and one is the predator, one is the hunter. So, the equations we have already seen look like this at any time instant x of h is the amount of hunter specimen and x of p is the amount of prey specimen. So, we will see how for a certain initial condition the solution looks like. So, this we are right now in scilab and we are going to execute a file very soon, I will show the code of the file also.

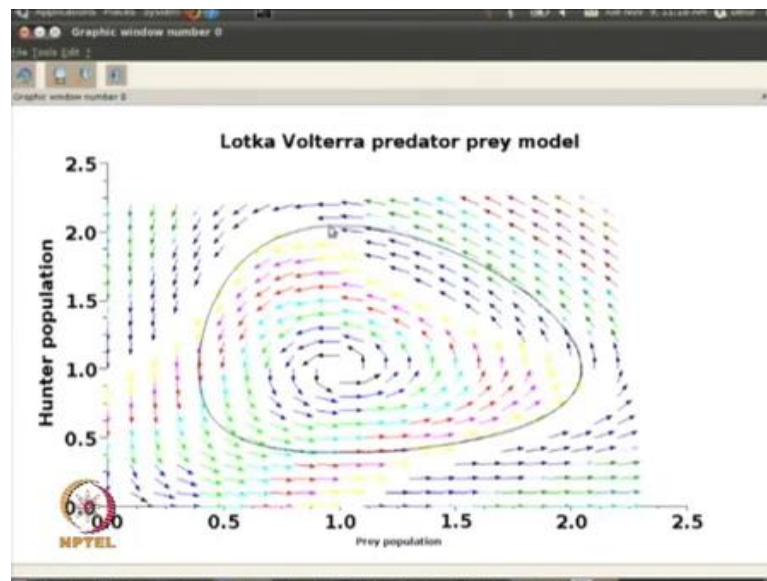
(Refer Slide Time: 01:37)



We will begin with executing file called Lotka main this calls another function called Lotka Volterra. So, when we execute this the first important thing we require is the vector field. So, the vector field as we can see consists of arrows at different points. We see that at this particular point at 1, 1, the point 1, 1 is the equilibrium point is an equilibrium point, while all around it. We have arrows suggesting that it is a center this particular plot can be obtained by using the champ command in scilab. Very soon we will open and see the scilab code that generates this, while this only indicates the vector field at different points, we will also like to see a trajectory, what is the trajectory? What is the solution to the differential equation? It is a curve which has these arrows precisely as the tangents at each point on the trajectory.

So, in order to see the trajectory, we have a trajectory here. Suppose, we start at some particular points, then we have this curve the arrows are not drawn on that particular curve, but the arrows we can see from the vector field y.

(Refer Slide Time: 02:51)



So, we can also change the initial condition and see, how the solution looks for a different initial condition. So, for that purpose we will modify the code and change the initial condition.

(Refer Slide Time: 03:17)

```
belur@madhu-laptop: ~/talks/scilab/code/Mani_Kochi_Feb10/madhu
File Edit View Terminal Help
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// Code to simulate the Lotka Volterra predator prey model

clf;
funcprot(0);
y0=[0.53; 0.53];
t=0:0.1:10;
t0=0;

exec('lotka_volterra.sci',2)

deff('vxf(x,y)', 'vx=(1-y)*x');
deff('vyf(x,y)', 'vy=(-1*x)*y');

x=linspace(0,2,23);
y=linspace(0,2,23);

vx=feval(x,y,vx);
vy=feval(x,y,vy);

"lotk@main.sce" 59L, 1123C 1,1 Top
```

So, let us have a look the code for this purpose, this is a code that was developed as a part of the talk to a teacher project at IIT Bombay, it is also being used, now for the NPTEL course. So, there are some initializations like clearing the figure the initial condition is being specified. Here, let us make the initial condition two. For example at 1, 1.

(Refer Slide Time: 03:25)

```
belur@madhu-laptop: ~/talks/scilab/code/Mani_Kochi_Feb10/madhu
File Edit View Terminal Help
// Code to simulate the Lotka Volterra predator prey model

clf;
funcprot(0);
y0=[0,0; 0.53];
t=0:0.1:10;
t0=0;

exec('lotka_volterra.sci',2)

deff('vxf(x,y)', 'vx=(1-y)*x');
deff('vyf(x,y)', 'vy=(-1*x)*y');

x=linspace(0,2,23);
y=linspace(0,2,23);

vx=feval(x,y,vx);
vy=feval(x,y,vy);

champ1(x,y,vx,vy)

// le... 'Proposed method', 'Current method');
xtitle('Lotka Volterra predator prey model');

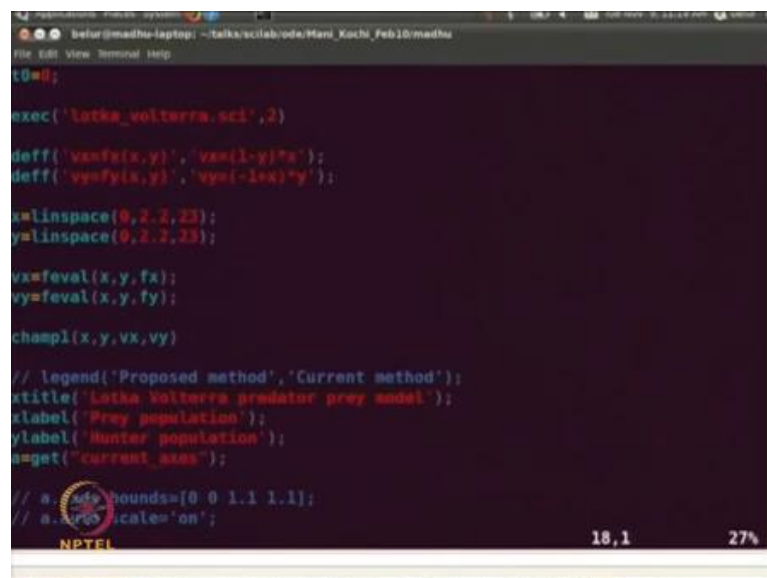
NPTEL 9,7 11%
```

This is where the initial condition is specified, and this is the time for which we are going to perform the simulation. The differential equation itself is being specified inside

this other file called Lotka Volterra dot sci, we will open and see that. Also the same differential equation is has also been specified here, for the purpose of the champ command. So, champ and champ one will together will draw the vector fields at different points.

We can specify how finely we want the vectors plotted, do we want very few vectors plotted at far points or we want a final grid of points at each point, we want the vector field the vector drawn. So, the main part up to drawing the vector field is up to here. Then, the plot also has some additional information like that it is, which axis corresponds to the prey population, which axis corresponding to the hunter population, those information are put here.

(Refer Slide Time: 04:18)



```
delur@madhu-laptop: ~/talks/scilab/ode/Mari_Kochi_Feb10/madhu
File Edit View Terminal Help
t0=0;

exec('lotka_volterra.sci',2)

deff('vnx(x,y)', 'vnx=(1-y)/x');
deff('vny(x,y)', 'vny=(-1-x)*y');

a=linspace(0,2,2,23);
y=linspace(0,2,2,23);

vx=feval(x,y,vx);
vy=feval(x,y,vy);

champ1(x,y,vx,vy)

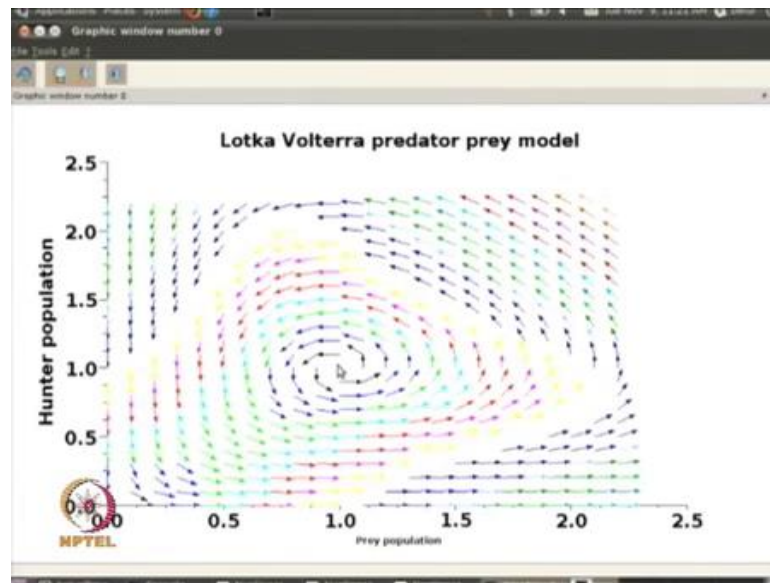
// legend('Proposed method','Current method');
xtitle('Lotka Volterra predator prey model');
xlabel('Prey population');
ylabel('Hunter population');
a=get('current_axes');

// a.axes.bounds=[0 0 1.1 1.1];
// a.axes.scale='on';

NPTEL 18,1 27%
```

And finally, the Lotka Volterra function file itself is being called from inside this particular function called ode. The function Odein scilab solves the ordinary differential equation with this initial condition and with this initial time, for up to this time duration. And finally, we are going to plot these both on the same graph on which we have drawn the vector field. So, let us go back we have given the initial condition now as this.

(Refer Slide Time: 05:23)



So, let us now run this code for that purpose, let us close this figure. So, we have again the vector field, now we see that there is no point, there is no trajectory why because the entire trajectory has collapsed to just one point, because we have given 1, 1 the equilibrium point itself as the initial condition.

(Refer Slide Time: 05:45)

```
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu
File Edit View Terminal Help
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// Code to simulate the Lotka Volterra predator prey model

clf;
funcprot(0);
y0=[1, 1];
t=0:0.1:10;
t0=0;

exec('lotka_volterra.sci',2)

deff('varfx(x,y)', 'x*(1-y)*a');
deff('varfy(x,y)', 'y*(1-x)*b');

x=linspace(0,2,23);
y=linspace(0,2,23);

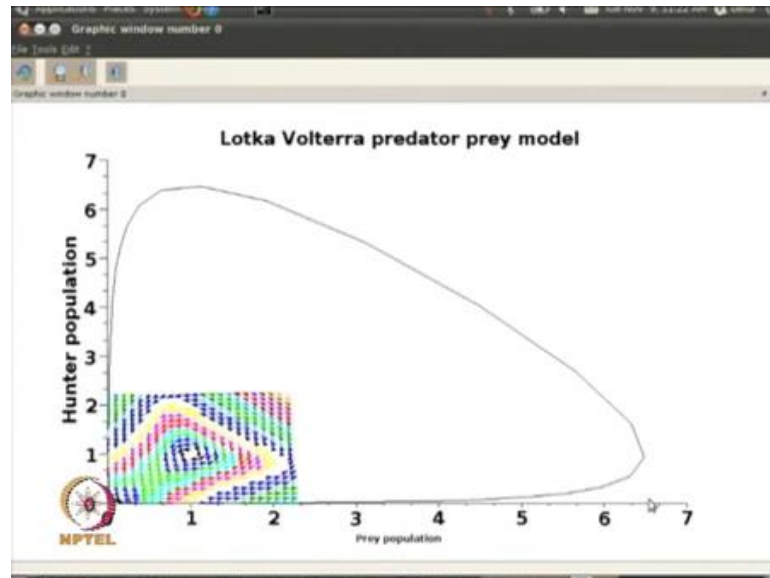
ux=feval(x,y,fx);
uy=feval(x,y,fy);

"lotka_main.sce" 59L, 1123C written          9,11          Top
```

So, as expected we do not expect to see any trajectory. The trajectory is a one point single point. So, let us now give some particular initial condition, which is very close to the axis, this is very close to the y axis, because the x component has been made to be

very small. So, it turned out that the arrows that we had drawn was for a very small region the trajectory itself goes through a very large area.

(Refer Slide Time: 06:09)



This is how the trajectory looks in fact, if we give a point on one of this axis. Then, we have already seen that it just blows up, if you have the hunter population equal to 0 due to some reason. Then, the prey population just goes on increasing exponentially and eventually becomes unbounded. So, here we see that by starting very close to the y axis also eventually we reach a similar situation.

(Refer Slide Time: 06:52)

```
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu
File Edit View Terminal Help
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems; code written by Madhu Belur.

// Code to simulate the Lotka Volterra predator prey model.

clf;
funcprot(0);
y0=[1.01; 1.01];
t=0:0.1:10;
t0=0;

exec('lotka_volterra.sci',0)

deff('vorf(x,y)', 'x=1-y!*x');
deff('vorf(y,x)', 'y=1-x!*y');

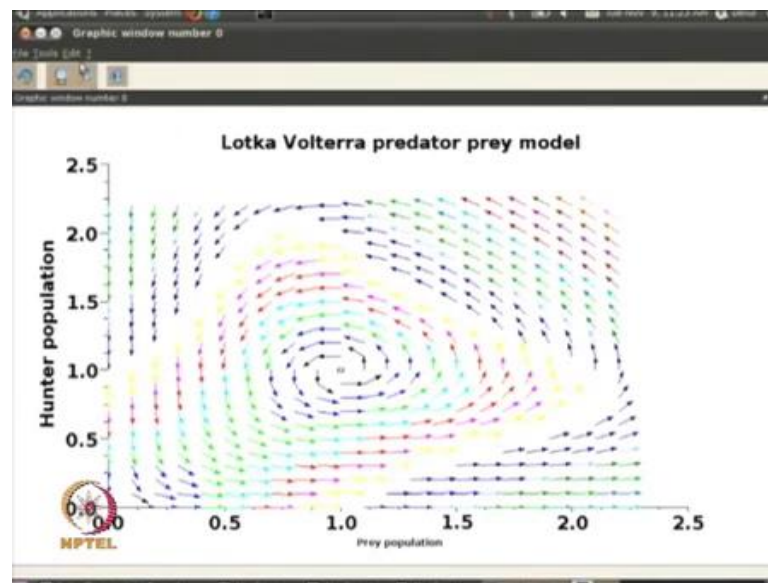
x=linspace(0,2,20);
y=linspace(0,2,20);

xx=feval(x,y,fx);
yy=feval(x,y,fy);

"lotka_main.sce" 59L, 1123C written          9,14          Top
```


We could now give an initial condition that is fairly close to the equilibrium point. So, here we have got this as the initial condition and it is, we see that there is a small circle. Here, it is a little too small compared to the rest of the region and hence it is not very visible.

(Refer Slide Time: 07:09)



We could consider zooming in and see this particular region. So, this is as far as the Lotka Volterra simulation is concerned.

(Refer Slide Time: 07:36)

```
beur@madhu-laptop: ~/talks/scilab/vde/Mani_Kochi_Feb10madhu
File Edit View Terminal Help
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// Code to simulate the Lotka Volterra predator prey model

clf;
funcprot(0);
y0=[1.01; 1.01];
t=0:1:10;
t0=0;

exec('lotka_volterra.sci',2)

deff('vxf(x,y)', 'vxf=(1-y)*x');
deff('vyf(x,y)', 'vyf=-1*x*y');

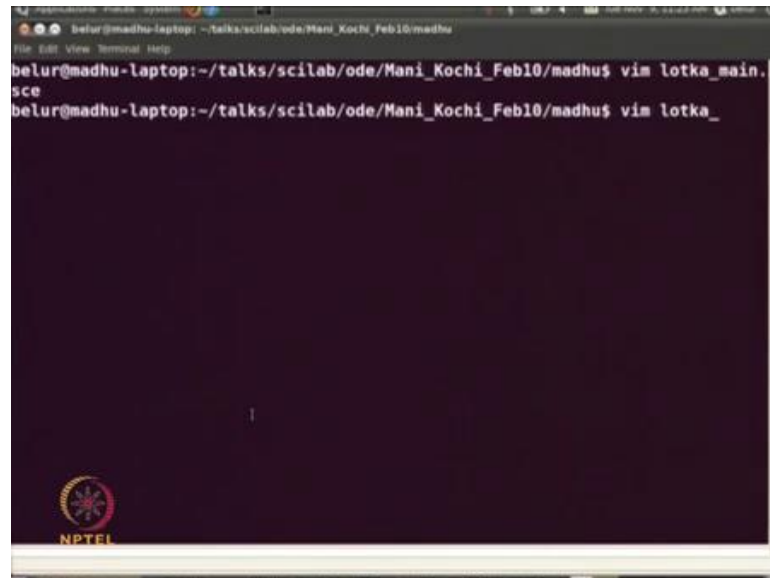
x=linspace(0,2,23);
y=linspace(0,2,23);

vx=feval(x,y,vxf);
vy=feval(x,y,vyf);

"lotka_main.sce" 59L, 1123C written          9,14      Top
```

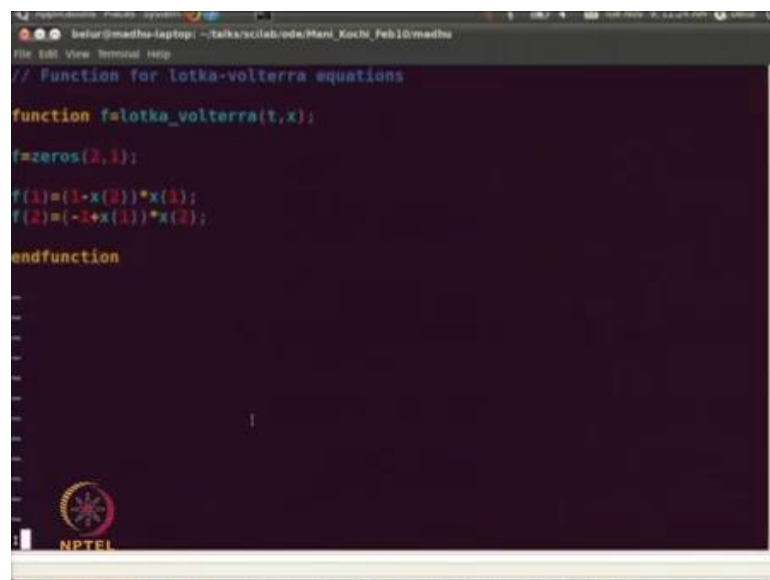

We were to also see this particular other this other file. So, this is just the function it defines given x, what is x dot is nothing but f of x. So, f is getting defined inside this function.

(Refer Slide Time: 07:48)



```
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim lotka_main.sce
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim lotka_
```

(Refer Slide Time: 07:51)



```
// Function for lotka-volterra equations
function f=lotka_volterra(t,x);
f=zeros(2,1);
f(1)=(1-x(2))*x(1);
f(2)=(-1+x(1))*x(2);
endfunction
```

So, it takes at any time t a value of x it gives us, what is f of x at that point. So, these equations are nothing but Lotka Volterra predator prey model, and we see that time does not play a role explicitly. Now, we will see the Van Der Pol oscillator for different initial

conditions, let recall that the Van Der Pol oscillator equations look like this, we had a second order differential equation.

(Refer Slide Time: 08:19)

```

belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim lotka_main.
sce
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim lotka_volte
rra.sci
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$
  
```

Which we had converted to a first order differential equation, we also saw an R L C circuit in which we could interpret x_1 as the voltage across the 3 components. The resistor inductor and the capacitor are all in parallel the resistor was not the simple resistor.

(Refer Slide Time: 08:19)

Existence of closed orbit

Consider the differential equation

$$\ddot{v} + \epsilon h(v)\dot{v} + v = 0$$

where $h(v) = -1 + v^2$.

Choose the state variables as $x_1 = v$ and $x_2 = \dot{v} + \epsilon H(v)$, where $H(v)$ is such that $\frac{d}{dv}H(v) = h(v)$ and $H(0) = 0$.

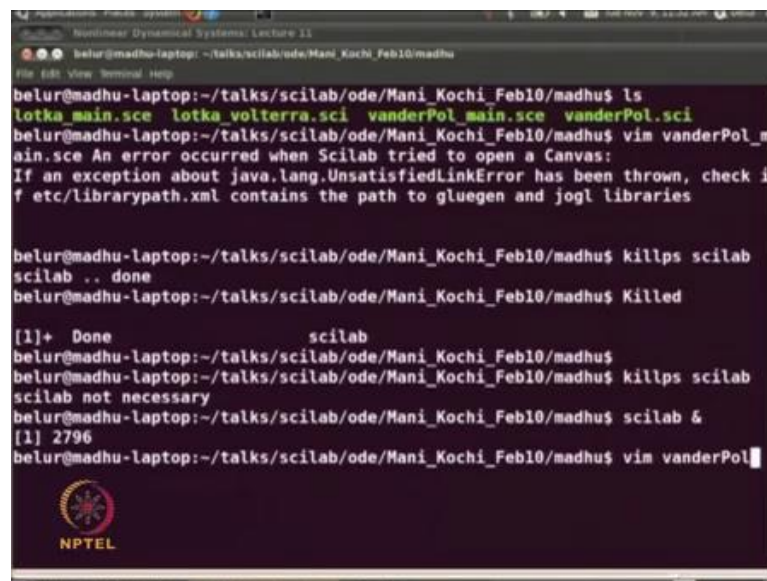
Therefore,

$$\begin{aligned} \dot{x}_1 &= x_2 - \epsilon H(x_1) \\ \dot{x}_2 &= -x_1 \end{aligned}$$

It has a unique equilibrium point, which is at the origin.

We had seen, it was a special one, which could either absorb or give energy in which sense it was a special, it was a active device. So, x_1 was the voltage across these 3 devices, which were connected in parallel and x_2 could be given the interpretation that it is the current through the inductor. Then, this second order differential equation can be considered as this in which h is a little special, because it denotes the characteristics of the resistance.

(Refer Slide Time: 09:31)



```
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ ls
lotka_main.sce lotka_volterra.sci vanderPol_main.sce vanderPol.sci
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce An error occurred when Scilab tried to open a Canvas:
If an exception about java.lang.UnsatisfiedLinkError has been thrown, check i
f etc/librarypath.xml contains the path to gluegen and jogl libraries

belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab .. done
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ Killed

[1]+ Done scilab
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab not necessary
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ scilab &
[1] 2796
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol
```

So, let us see a simulation of this particular model, so like before we have this particular scilab code already written this. Also, was written as a part of the talk to a teacher project at IIT Bombay and it is being also used for the NPTEL course.

(Refer Slide Time: 09:37)

```
Nonlinear Dynamical Systems, Lecture 11
belur@madhu-laptop: ~/talks/scilab/nde/Mani_Kochi_Feb10/madhu
File Edit View Terminal Help

// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// code to simulate the van der Pol oscillator

clf;
funcprot(0);
y0=[4.53; 4.53];
t0=0.1:100;
t0=0;
exec('vanderPol.sci',2)
epsilon=0.1;

deff('vx=fv(x,y)', 'vxy-epsilon*(1-x*x.^3)');
deff('vy=fy(x,y)', 'vy=-x');

x=linspace(-5,5,11);
y=linspace(-5,5,11);
vx=fv(x,y);
vy=fy(x,y);
"vanderPol_main.sce" 63L, 1144C                               10,1   Top
```

So, we have the initial condition being specified here the function itself for the differential equation is being specified inside another file called Van Der Pol dot sci, which we will open and see in a minute.

(Refer Slide Time: 09:55)

```
Nonlinear Dynamical Systems, Lecture 11
belur@madhu-laptop: ~/talks/scilab/nde/Mani_Kochi_Feb10/madhu
File Edit View Terminal Help

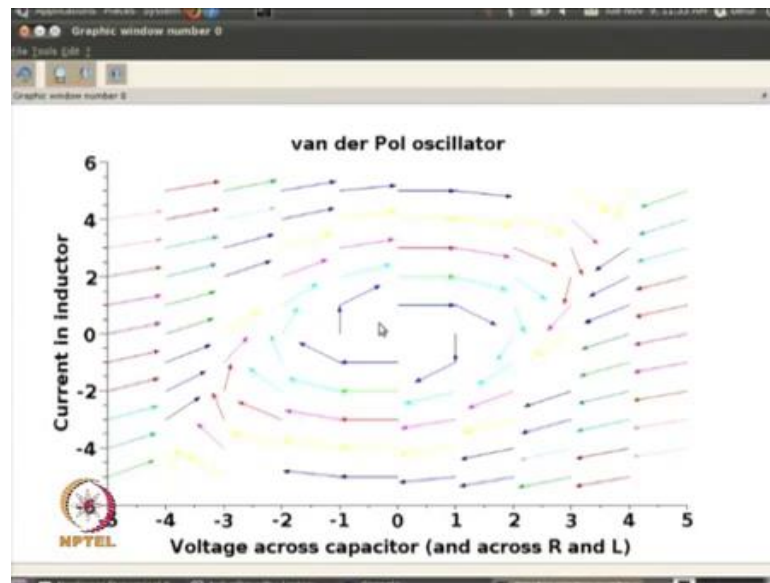
clf;
funcprot(0);
y0=[4.53; 4.53];
t0=0.1:100;
t0=0;
exec('vanderPol.sci',2)
epsilon=0.1;

deff('vx=fv(x,y)', 'vxy-epsilon*(1-x*x.^3)');
deff('vy=fy(x,y)', 'vy=-x');

x=linspace(-5,5,11);
y=linspace(-5,5,11);
vx=fv(x,y);
vy=fy(x,y);
champ(x,y,vx,vy)
"vanderPol_main.sce" 63L, 1144C                               12,1   12%
```

So, first we are interested in seeing the vector field, so for that purpose let us run this. So, this is how the arrows look, we have the origin 0, 0 as the equilibrium point.

(Refer Slide Time: 10:10)



The only equilibrium point, and we have these arrows going all round suggesting that it is the center whether it is a center or not. We can come to know only by linearizing and seeing the equilibrium by seeing the Eigen values at this, which we had already seen at the origin is unstable equilibrium point, but these arrows are not telling us much information. We will also see depending on whether, we start from outside or certain limit cycle or inside these arrows are going to tell whether, we will be converging to the limit cycle either from the inside or the outside.

So, let us we have already specified an initial condition first, we have seen only the arrows and it is waiting for our a button to be pressed. So, for this particular initial condition the trajectories are coming closer and closer and encircling round and round, and eventually converging to this limit cycle. So, that is for the case that, we start from this particular point, we can take another two points also from outside this particular limit cycle.

(Refer Slide Time: 11:34)

```
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// code to simulate the van der Pol oscillator

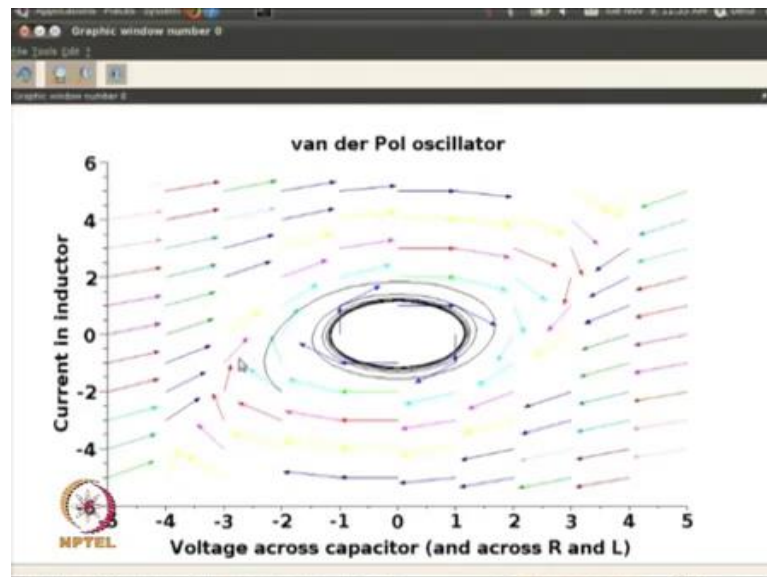
clf;
funcprot(0);
y0=[-2;-1.5];
t=0:0.1:100;
t0=0;
exec('vanderPol.sci',2);
epsilon=0.1;

deff('vx=fx(x,y)','vxy=epsilon*(-xx.^2)');
deff('vy=fy(x,y)','vym=-x');

x=linspace(-5,5,11);
y=linspace(-5,5,11);
vx=fvanderPol(y,fx);
```

So, the limit cycle is seen to be going between minus 1 and 1 on the horizontal axis. And between 1 and 1 on the vertical axis also, let us take another initial condition. Let us take, minus 2 and minus 2 and run this program again. So, we have just the vector field now.

(Refer Slide Time: 12:00)



So, we have right now the initial condition starting from here and it is encircling like this and converging to the limit cycle, we can take a point. Let us say, here corresponding to 3, minus 4, and see whether this also encircles, this also expected it would not take long.

(Refer Slide Time: 12:22)

```
belur@madhu-laptop: ~/talks/scilab/vids/Main_Kochi_Feb10/madhu
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// code to simulate the van der Pol oscillator

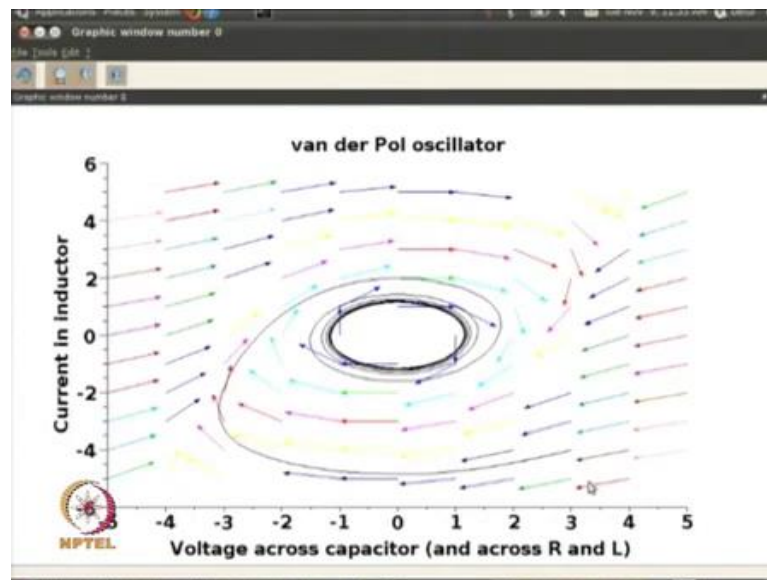
clf;
funcprot(0);
y0=[3;-2];
t=0:0.1:100;
t0=0;
exec('vanderPol.sci',2);
epsilon=0.1;

deff('vx=fx(x,y)', 'vxy-epsilon*(1-x*x.^2)');
deff('vy=fy(x,y)', 'vym-x');

x=linspace(-5,5,11);
y=linspace(-5,5,11);
vx=fx(x,y);
vy=fy(x,y);
"vanderPol_main.sce" 63L, 1139C written          9,8      Top
```

To check 3, minus 4 the vector field after having started here, we see this particular trajectory. So, we see that the arrows are all tangent to this particular trajectory, we will now take a point from inside the limit cycle first, we could take just the origin.

(Refer Slide Time: 12:32)



We know that the origin being equilibrium points, we do not expect to see any trajectory and indeed, we do not see any trajectory. So, there is actually just one very thin point, which is the entire trajectory because that is the equilibrium point.

(Refer Slide Time: 12:47)

```
Scilab - Console
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// NonLinear dynamical systems: code written by Madhu Belur

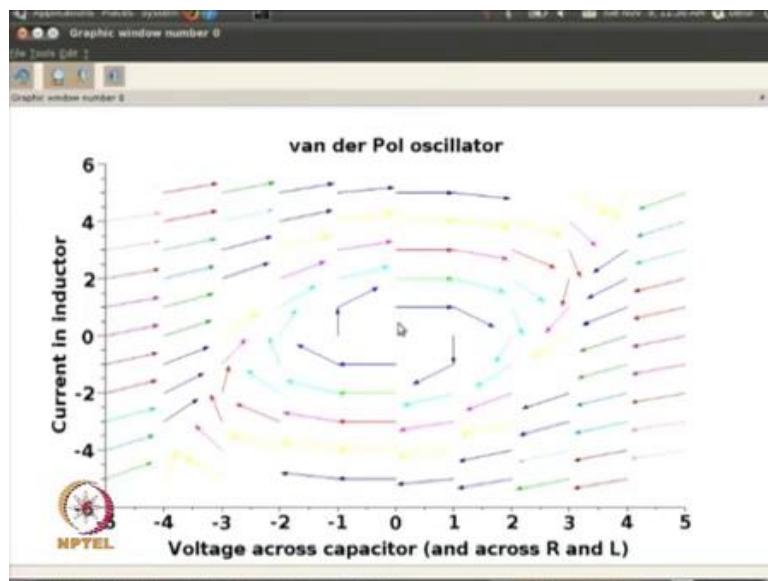
// code to simulate the van der Pol oscillator

clf;
funcprot(0);
y0=[0;-1];
t=0:0.1:100;
t0=0;
exec('vanderPol.sci',2)
epsilon=0.1;

deff('vx=fx(x,y)','vxy-epsilon*(1-x*x.^2)');
deff('vy=fx(x,y)','vym-x');

x=linspace(-5,5,11);
y=linspace(-5,5,11);
vx=fx(x,y);
vy=fx(x,y);
"vanderPol_main.sce" 63L, 1138C written          9,8      Top
```

(Refer Slide Time: 12:55)



(Refer Slide Time: 13:12)

```
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// code to simulate the van der Pol oscillator

clf;
funcprot(0);
y0=[0;0.5];
t=0:0.1:100;
t0=0;
exec('vanderPol.sci',2);
epsilon=0.1;

deff('vx=fx(x,y)', 'vxy-epsilon*(1-x*x.^2)');
deff('vy=fy(x,y)', 'vym-x');

x=linspace(-5,5,11);
y=linspace(-5,5,11);
vx=fv(x,y,vx);
vy=fv(x,y,vy);
"vanderPol_main.sce" 63L, 1139C written          9.9      Top
```

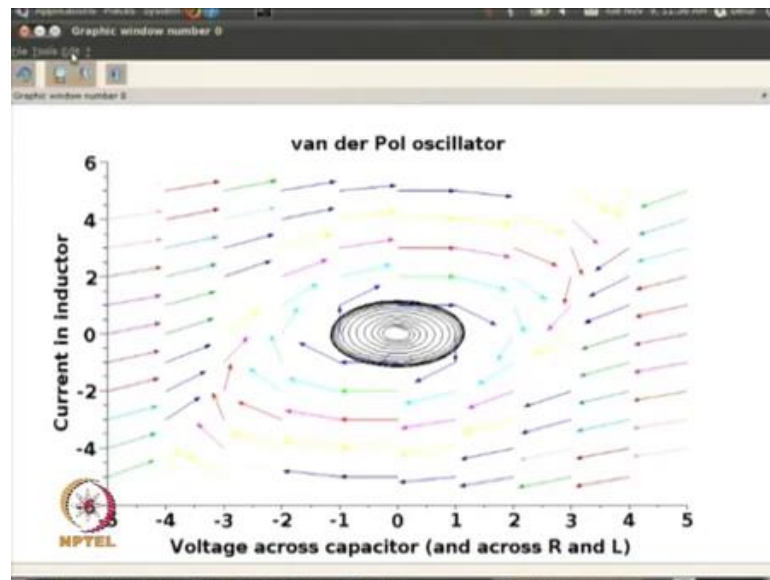
So, let us take a point also inside this limit cycle, but not the origin. Let us say, we can take point two here is a vector field. So, here let us again blow this up.

(Refer Slide Time: 13:20)

```
Console
File Edit Preferences Control Applications ?
-->y=linspace(-5,5,11);
-->vx=fval(x,y,fx);
-->vy=fval(x,y,fy);
-->champ1(x,y,vx,vy)
-->// legend('Proposed method','Current method');
-->xtitle('van der Pol oscillator');
-->ylabel('Current in inductor');
-->xlabel('Voltage across capacitor (and across R and L)');
```

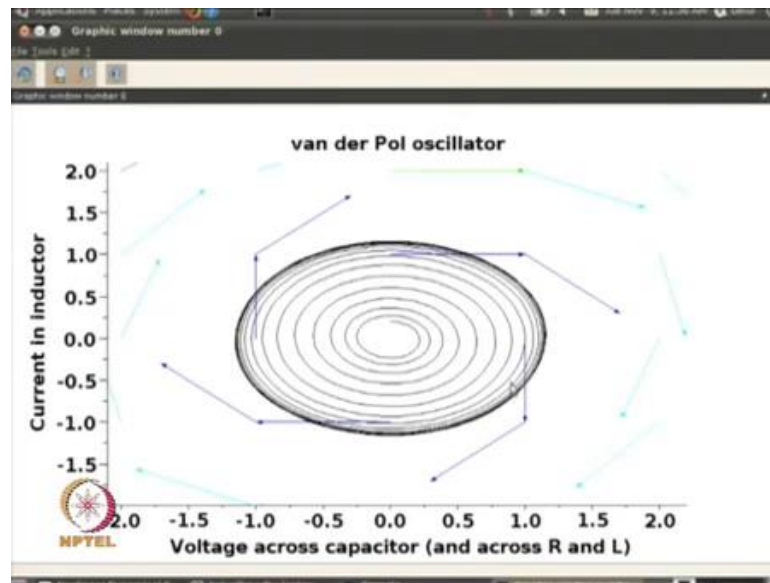
So, here when we start from here it seems to encircle and go outside and outside and eventually converge to this limit cycle.

(Refer Slide Time: 13:23)



So, let us open and see the file Van Der Pol dot sci, because that is being called by the ode function in scilab.

(Refer Slide Time: 13:31)



So, Van Der Pol dot sci is just defining the first component in x dot as x minus 2 epsilon times $h \times 1 H$.

(Refer Slide Time: 13:41)

```
Console
Belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu
File Edit View Terminal Help
// Scilab code developed as part of the Talk to a teacher project at
// IIT Bombay. Code also used later for NPTEL course on
// Nonlinear dynamical systems: code written by Madhu Belur

// code to simulate the van der Pol oscillator

clf;
funcprot(0);
y0=[0;0.2];
t=0:0.1:100;
t0=0;
exec('vanderPol.sci',2)
epsilon=0.1;

deff('vx=fx(x,y)', 'vx=y-epsilon*(-x*x.^3)');
deff('vy=fy(x,y)', 'vy=-x');

x=linspace(-5,5,11);
y=linspace(-5,5,11);
vx=fx(x,y,t);
NPTEL
"vanderPol_main.sce" 63L, 1139C written 9,9 Top
```

(Refer Slide Time: 13:47)

```
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ ls
lotka_main.sce  lotka_volterra.sci  vanderPol_main.sce  vanderPol.sci
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce An error occurred when Scilab tried to open a Canvas:
If an exception about java.lang.UnsatisfiedLinkError has been thrown, check i
f etc/librarypath.xml contains the path to gluegen and jogl libraries

belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab .. done
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ Killed

[1]+  Done                  scilab
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab not necessary
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ scilab &
[1] 2796
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol
```

(Refer Slide Time: 13:52)

```
// Function for lotka-volterra equations

function f=vanderPol(t,x);

epsilon=0.1;
f=zeros(2,1);

Hx1=-x(1)+x(1)^3;
f(1)=x(2)-epsilon*Hx1;
f(2)=-x(1);

endfunction

NPTEL
"vanderPol.sci" 13L, 165C 8,5 All
```

(Refer Slide Time: 14:09)

Existence of closed orbit

Consider the differential equation

$$\ddot{v} + \epsilon h(v)\dot{v} + v = 0$$

where $h(v) = -1 + v^2$.
Choose the state variables as $x_1 = v$ and $x_2 = \dot{v} + \epsilon H(v)$, where $H(v)$ is such that $\frac{d}{dv}H(v) = h(v)$ and $H(0) = 0$.
Therefore,

$$\begin{aligned}\dot{x}_1 &= x_2 - \epsilon H(x_1) \\ \dot{x}_2 &= -x_1\end{aligned}$$

It has a unique equilibrium point, which is at the origin.

NPTEL

x_1 itself was a function, which we had seen in the slide. It is equal to, this is h of v and it is integral with h of 0 equal to 0. That is how capital h is defined that defines capital H uniquely and that evaluated x_1 is, what capital H of x_1 is that is being defined.

(Refer Slide Time: 14:29)

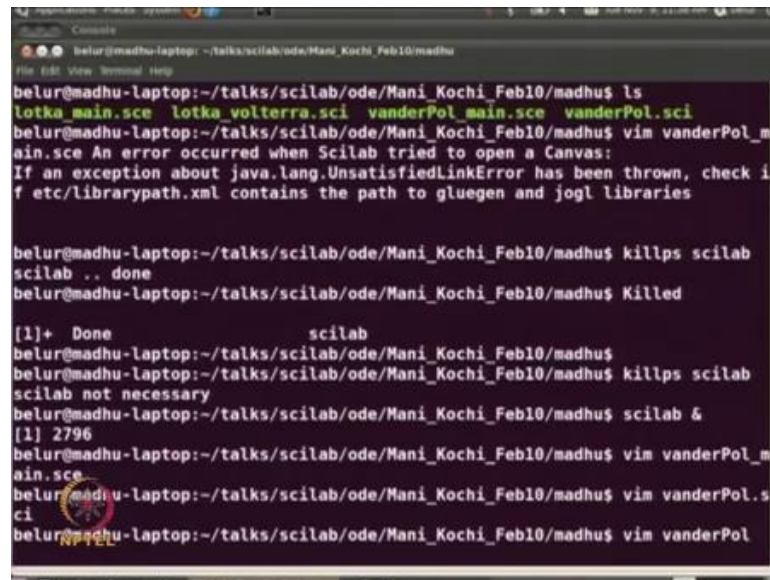
```
// Function for lotka-volterra equations
function f=vanderPol(t,x);
epsilon=0.1;
f=zeros(2,1);
Hx1=-x(1)+x(1)^2;
f(1)=x(2)-epsilon*Hx1;
f(2)=-x(1);
endfunction
```

NPTEL
"vanderPol.sci" 13L, 165C

Here, that is being put in here and that defines the first component of \dot{x} , second component of \dot{x} is nothing but x_1 . So, the output of this particular function file is f_1 and f_2 , which denote nothing but \dot{x}_1 and \dot{x}_2 this is going to be utilized by the ode command in scilab. So, this explains, how we can use scilab to visualize the

dynamics of a differential equation while a second order differential equation can be visualized using the vector field. Also by using the champ and champ 1 command more generally. We can use ode command to see to obtain the solution to a differential equation and initial value problem in particular.

(Refer Slide Time: 15:13)



```
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ ls
lotka_main.sce  lotka_volterra.sci  vanderPol_main.sce  vanderPol.sci
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce An error occurred when Scilab tried to open a Canvas:
If an exception about java.lang.UnsatisfiedLinkError has been thrown, check i
f etc/librarypath.xml contains the path to gluegen and jogl libraries

belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab .. done
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ Killed

[1]+  Done                  scilab
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab not necessary
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ scilab &
[1] 2796
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol.s
ci
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol
```

So, let us like we did for the Lotka Volterra scilab code, we will also see the scilab code completely. Of course, both the codes all the 4 scilab programs will be made available on the website, it is not required to copy this from this screen. It will be made available completely, this part of the code is to set parameters of the plot all these commands can be obtained, can be seen in detail, can be used more effectively by using the help command in scilab as far as we are concerned these codes will be made available, these completes this topic.

(Refer Slide Time: 15:56)

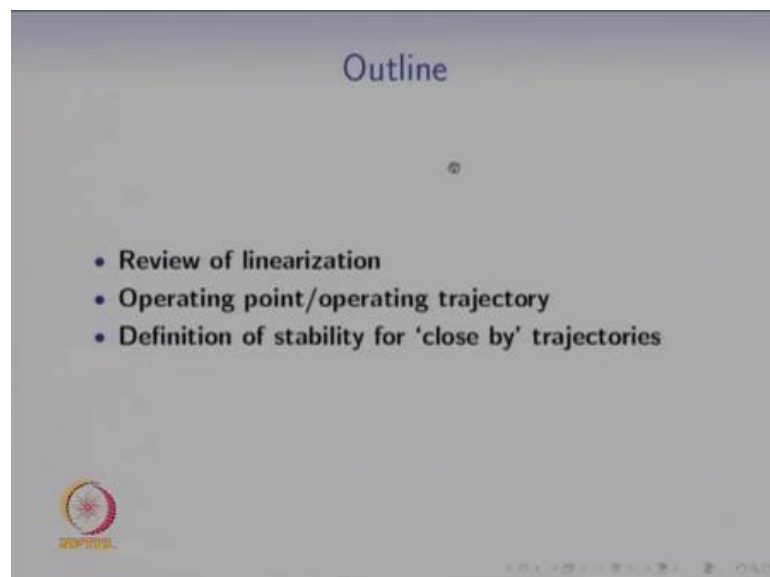
```
belur@madhu-laptop: ~/talks/scilab/ode/Mani_Kochi_Feb10/madhu
lotka_main.sce lotka_volterra.sci vanderPol_main.sce vanderPol.sci
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce An error occurred when Scilab tried to open a Canvas:
If an exception about java.lang.UnsatisfiedLinkError has been thrown, check i
f etc/librarypath.xml contains the path to gluegen and jogl libraries

belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab .. done
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ Killed

[1]+ Done scilab
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ killps scilab
scilab not necessary
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ scilab &
[1] 2796
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol.s
ci
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$ vim vanderPol_m
ain.sce
belur@madhu-laptop:~/talks/scilab/ode/Mani_Kochi_Feb10/madhu$
```

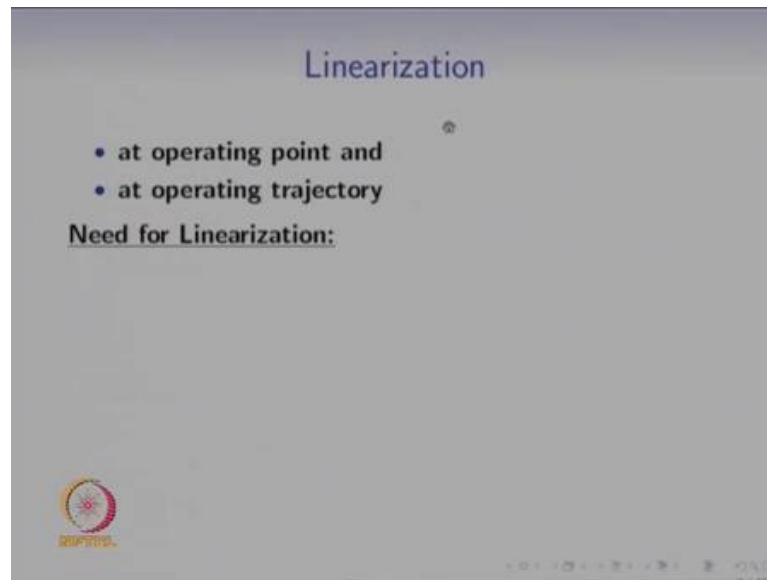
We will continue on another topic now, we now continue to the next topic. So, we are also going to see something about linearization about a trajectory, we have already seen linearization about a point. Now, we will also see linearization about a operating trajectory the significance of that why it is important to study that is, what we will begin seeing.

(Refer Slide Time: 16:17)



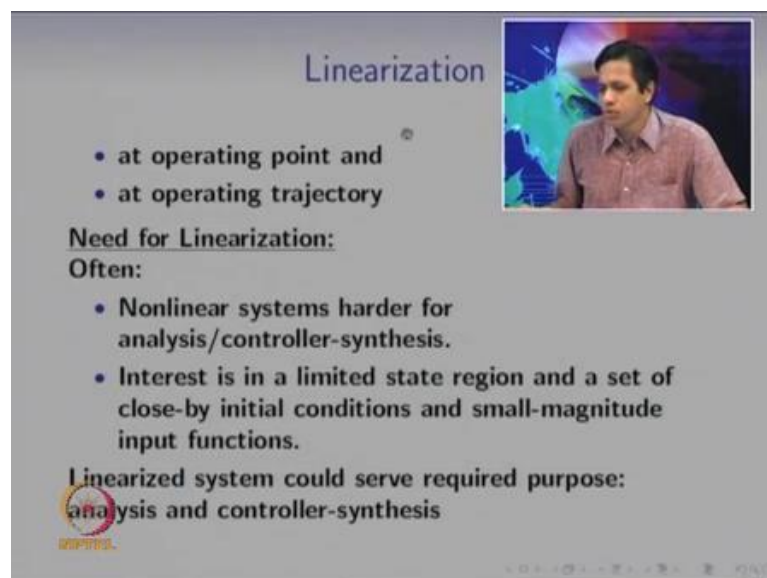
So, we will quickly review linearization after that, we will see what the meaning of operating point and operating trajectory is? We will also see a definition of stability the notion of stability of close by trajectories.

(Refer Slide Time: 16:38)



So, we when we speak of linearization till we spoke about linearization about an operating now, when we say linearization, we will distinguish in future between linearization about an operating point, and linearization about an operating trajectory. So, let us quickly see, what is the need for linearization.

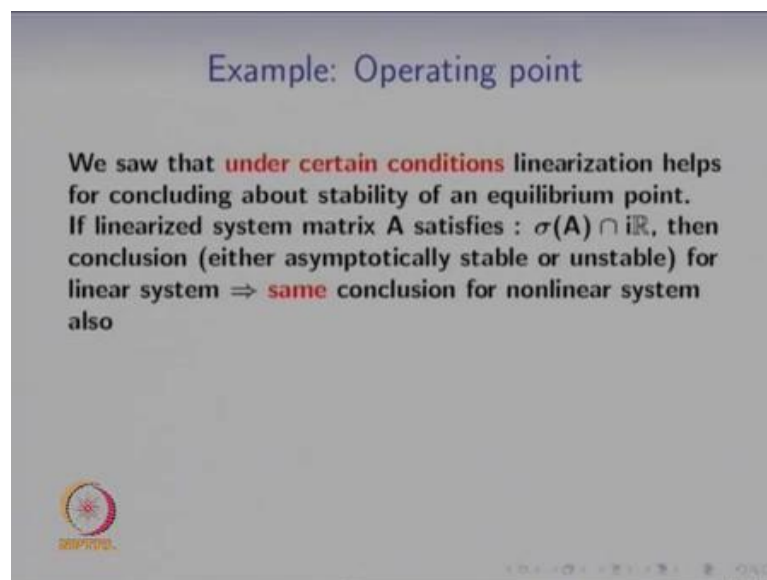
(Refer Slide Time: 17:41)



So, while we are studying non-linear systems, it is acknowledged that non-linear systems are harder both for analysis and for controller-synthesis, but it is also true that the interest of the analysis of the controller design is to a limited region of the state. And also to a small set of initial conditions that are close by close to a certain important point. Also, the import functions may not be of very large amplitude.

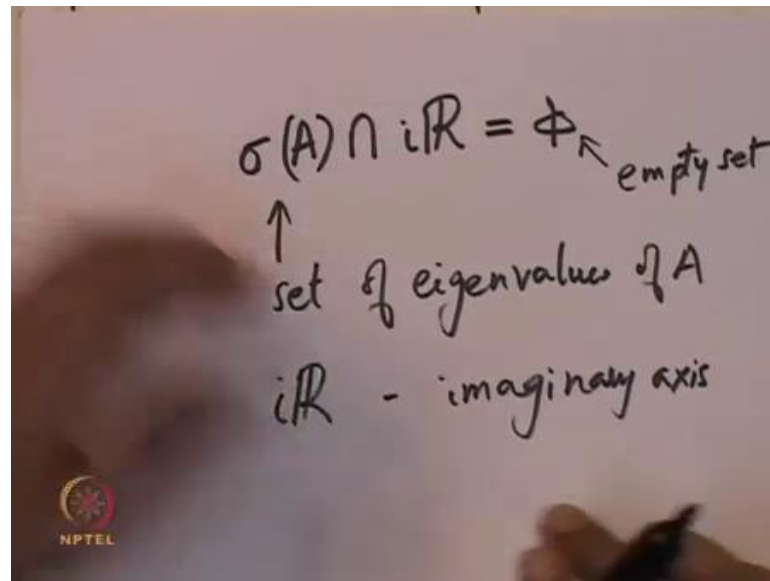
So, given these situations it is often helpful to linearize. So, the linearized system could serve the required purposes, which purposes the analysis of the non-linear system, and controller synthesis for the non-linear systems these both purposes is perhaps met by the linearized system. So, to what extent it is met we have only partially seen, we already saw that under certain conditions.

(Refer Slide Time: 18:03)



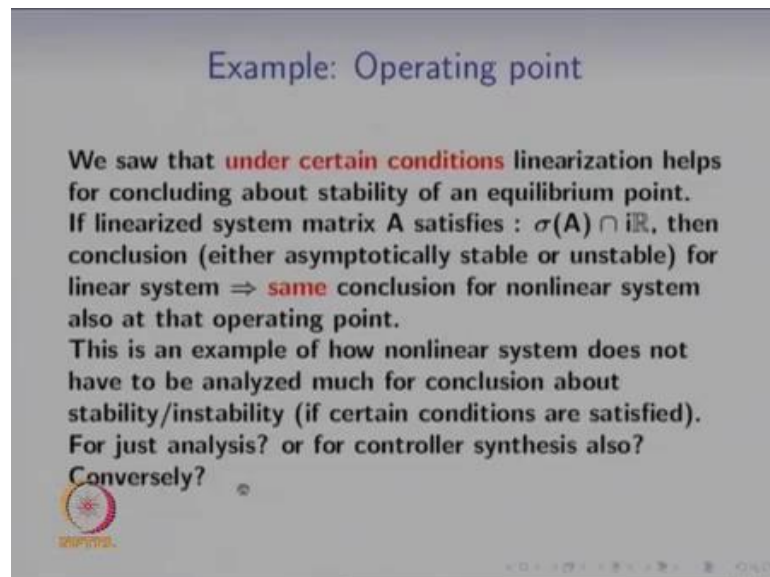
We will quickly review these conditions linearization helps for concluding about stability of an equilibrium point more precisely, if the linearized system, when we are speaking about an equilibrium point. We take the non-linear system and linearize, it about the equilibrium point the linearized system matrix **A**. We take if it satisfies this condition, what is this condition the set of Eigen values of a intersection of that set with this imaginary axis is empty, is empty I have missed writing. Here, I will just write this condition.

(Refer Slide Time: 18:45)



Here, suppose this condition is satisfied, what is the meaning of this set of Eigen values of A? And $i\mathbb{R}$ is the imaginary axis is the intersection of these two sets is the empty set meaning, in this context it means that there are no Eigen values of a on the imaginary axis, if this particular condition is satisfied.

(Refer Slide Time: 19:32)



So, this equal to phi is, what has been missed on the slide here, it has been missed here. So, if the intersection of the imaginary, if the intersection of the Eigen values of A, and the imaginary axis is empty. Then, the conclusion, which conclusion whether the system

is asymptotically stable or unstable that conclusion for the linear system also implies the same conclusion for non-linear system.

This is one of the most important results in the context of linearization of non-linear systems about that operating point, because non-linear system could have many equilibrium points. And we could consider the linearization about different operating points as far as the particular equilibrium point is concerned, we can obtain this matrix A , and if there are no Eigen values of A on the imaginary axis. Then, the linear system, the linearized system may be unstable or may be asymptotically stable. For example, it is asymptotically stable, if all the Eigen values of A are in the left half open, left half complex plane. It is unstable, if there is one or more Eigen values of A in the right half complex plane.

So, this particular conclusion on the linearized system, we are able to do using the linearized system matrix A that conclusion is the same for the non-linear system. Also about the equilibrium point under this condition under, which condition under the condition that there are no Eigen values of A on the imaginary axis. So, what is the significance of this is an example of how non-linear system does not have to be analyzed much as far as this particular conclusion goes as far as the conclusion, whether the non-linear system about an equilibrium point is unstable or asymptotically stable as far as that conclusion goes, it is enough to study the linearized system.

This conclusion is indeed the same only under certain conditions only, when certain conditions are satisfied, if those conditions are not satisfied. Then, it is not possible to say that the same conclusion holds; now we can ask is this something that is just for analysis or is this going to help for controller synthesis also. Is it that such a result can also be utilized for controller synthesis? This is something that we will see in the detail in the next few lectures. Another important thing is, what about converse, if you know that the non-linear system is unstable, is it true that you should the linearized.

System is also unstable that is the other question, when what we refer to here as conversely. So, these are important questions that we will address in the next few lectures; we will address the notion of controller synthesis for the linearized system, and whether that controller will work for the non-linear system also.

(Refer Slide Time: 22:43)

Controller synthesis for operating point

Controller synthesis for stabilizing at an operating point.

The to-be-controlled nonlinear system gives to-be-controlled linearized system.

Obtain A, B matrices for linear systems.

Consider $\dot{z} = Az + Bv$ (System Lin)

coming from $\dot{x} = f(x, u)$ (System Nonlin)

Under what conditions controller designed for System-Lin works for System-Nonlin also.

So, let us come back to this question controller synthesis for stabilizing a non-linear system at an operating point as I said, when we say, operating point. We need to know, distinguish that operating trajectory, and operating point in the context of linearization. So, let us first ask this question controller synthesis for stabilizing a non-linear system about an operating point. So, that to be controlled non-linear system gives A to be controlled.

Linearized system upon linearization the non-linear system, which had some inputs will give us a linearized system also with some inputs. So, suppose A and B are matrices for the linear system, we will do this in a little more precise way in the next few lectures. I am just motivating how a controller for the linearized system may work for the non-linear system also. Suppose, A and B are matrices for the linearized system.

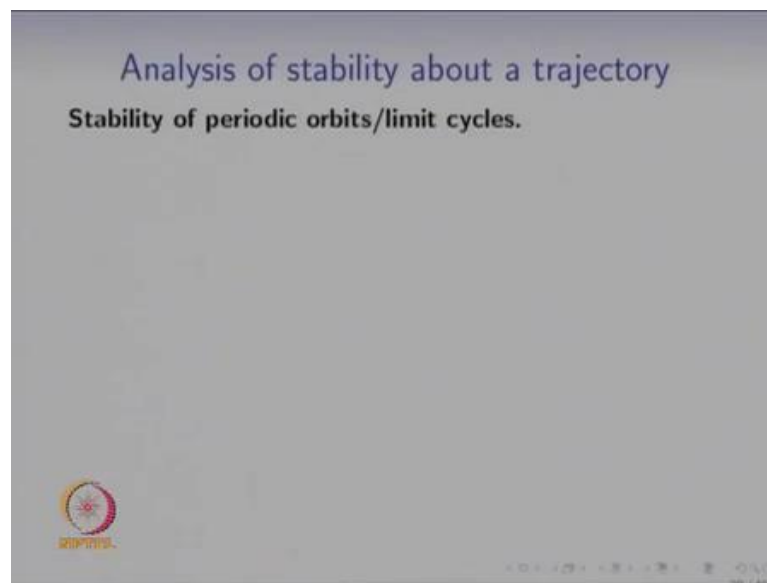
Let us call system L in and these matrices A and B are coming from the non-linear system \dot{x} is equal to f of x, u till now we had been studying only autonomous systems; there was no input u nor v. Now, we are speaking of a system which allows you to put a control input.

So, the linearized system is this in which the state has been called z \dot{x} is equal to a z plus b v and the original non-linear system was \dot{x} is equal to f of x, u. We will see, how A and B matrices are to be obtained from this particular function f under what

conditions? So, what is the controller synthesis question under what conditions controller designed for system L in also works for system non- L in the controller?

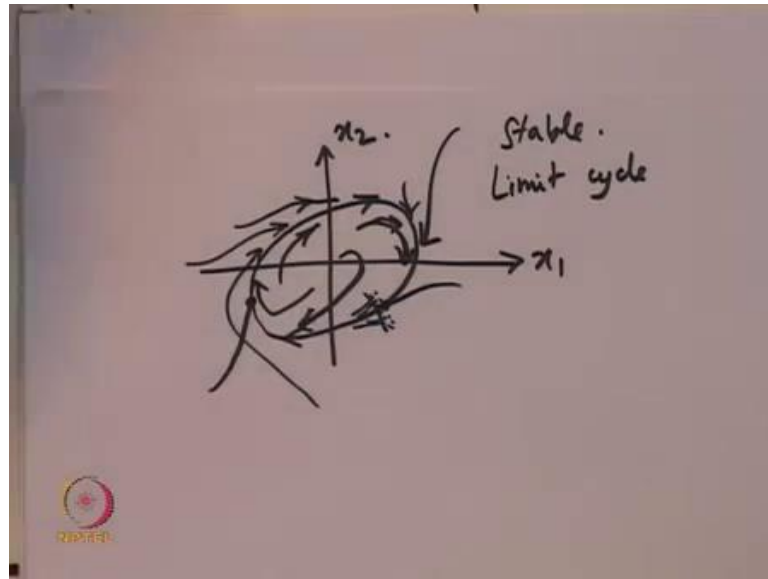
We have designed for this system under, what condition it will work for this system. Also given that A and B where obtained from these from this non-linear system may be, it is possible linear control theory for this system might automatically hold for this system also, in order to answer this question.

(Refer Slide Time: 24:49)



We have to understand in little more detail about linearization about an operating trajectory in the context of linearization. We will also speak about stability of the operating trajectory. So, we have already seen for a autonomous system, what it means for stability of an equilibrium point, we also Lyapunov stability of the equilibrium point. Now, the next question is what is the meaning of stability of a operating trajectory of a trajectory. So, let us first consider the case when that trajectory is a periodic orbit.

(Refer Slide Time: 25:30)



So, suppose this is a periodic orbit and now, we know that if we start on this periodic orbit. Then, we will keep going along this orbit, we could ask the question is this periodic orbit stable in the sense that trajectories that start close to this periodic orbit do they converge to this periodic orbit this is the question. We ask in the context of Van Der Pol oscillator. For example, we ask whether this periodic orbit is a stable limit cycle, what does it mean to be a limit cycle.

We have these other trajectories that are converging to this and stable limit cycle means that we take such a cut and we look at all initial conditions close to this periodic orbit. We could ask are all these trajectories, which trajectories all the trajectories that are starting close to this periodic orbit are all of them going to converge to this periodic orbit. That is ideally the case that is how we want for robust sustained oscillations of an oscillator that we build in a in a laboratory.

We want that no matter what initial conditions it should converge to that periodic orbit, because that periodic orbit perhaps is carefully designed to have the right period and right amplitude. So, this is what we will say a limit cycle a stable limit cycle a stable limit cycle is one in which we have a periodic orbit the cycle itself and trajectories that are close by come closer and closer to this periodic orbit. Of course, we know that it cannot come and intersect at any particular point, why is it that the two trajectories cannot Lipchitz at this point, if the function is Lipchitz.

Then, we know that the two trajectories cannot intersect. So, we have all trajectories that are coming close and closer to the stable limit cycle also from the inside when they start. So, we have already seen results in this context. So, we already know what it means for a trajectory to be stable? We informally know it for the purpose of this limit cycle for the case of periodic orbit.

(Refer Slide Time: 28:17)

Analysis of stability about a trajectory

Stability of periodic orbits/limit cycles.
Non-periodic orbits. For example:

- Suppose for a nonlinear system, optimal trajectory computed by optimal control.
- Say: minimum fuel consumption trajectory, or minimum time trajectory

Stability of a trajectory:

- Optimal input for staying on the optimal trajectory pre-decided by intensive computation. 'Open-loop control input pre-decided'
- Uncertainties perhaps cause difficulty in 'implementation'.

Do trajectories that are 'close by' stay 'close by'.

We already speak about stability of periodic orbits what about non-periodic orbits is that relevant for periodic orbits in the case of autonomous systems? We have already seen for oscillators it is very important, but for non-periodic orbits also should we study this? So, we have an example here. For example, suppose for a non-linear system the optimal trajectory was computed by optimal control.

So, we have a whole nice theory about optimal control and using that theory, suppose we compute the optimal trajectory for a particular system, this optimal might be. For example, it is the trajectory when the fuel consumed is minimum it might be a minimum fuel consumption trajectory for a rocket going up into the space, this is a very non-linear system many inputs many outputs. This rocket could be taken from the ground to the space and by using lot of computation may be we compute a trajectory by which least amount fuel is consumed in taking the rocket from the ground up to the space.

So, this particular trajectory computation can be done offline, we assume this or for example, this trajectory might have been. So, called a minimum time trajectory it might

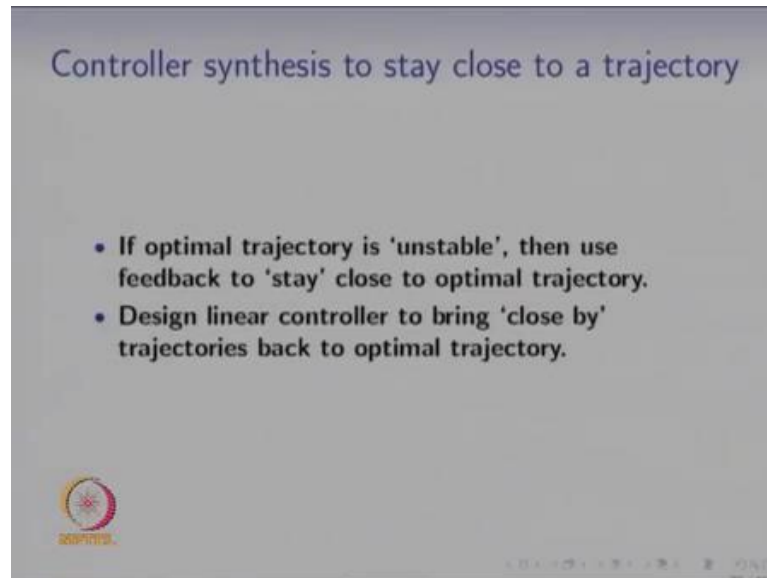
be the trajectory that is going to assure us that this rocket going from ground to space into the particular periodic orbit takes least amount of time it might be a minimum time trajectory. So, we assume that this particular trajectory computation is done offline. It is done on a computer with lots of time allowed before that the rocket goes up in the air, we already have a trajectory that will take it into the space in least amount of time or may be in least fuel consuming way.

So, this is what is optimal control, and once we have found this optimal trajectory this optimal input. That is required for staying on the optimal trajectory is pre-decided. For example, by intensive computation now, we can ask given that this particular system is not going to exactly go along this optimal trajectory, before we go into that suppose the open loop control input is pre-decided the meaning of that the input that you should give.

So, that that performance objective is met in optimal way that input is pre-decided and you just give this input to the system. This is same as saying that the control input is open loop. It is open loop control input, and it is pre-decided, but it is also true that there are various uncertainties in the system. The fuel may not be of the correct quality that was assumed when you were deciding, when you were computing the control input. There may be other uncertainties in the space, when the rocket is going up, it is encountering a situation that is not exactly accounted in the model.

So, because of these uncertainties it is difficult to implement exactly that particular optimal trajectory using this pre-decided control input that time. We could ask the question, what if the trajectories start close by. If, the trajectories that are close to this optimal trajectory do they stay close by. This is a natural question that arises in this context. If, the close by trajectories do not stay close by, if the optimal trajectory is good, but the trajectories that are close to the optimal trajectory perhaps are moving away from the system away from the optimal trajectory.

(Refer Slide Time: 31:58)

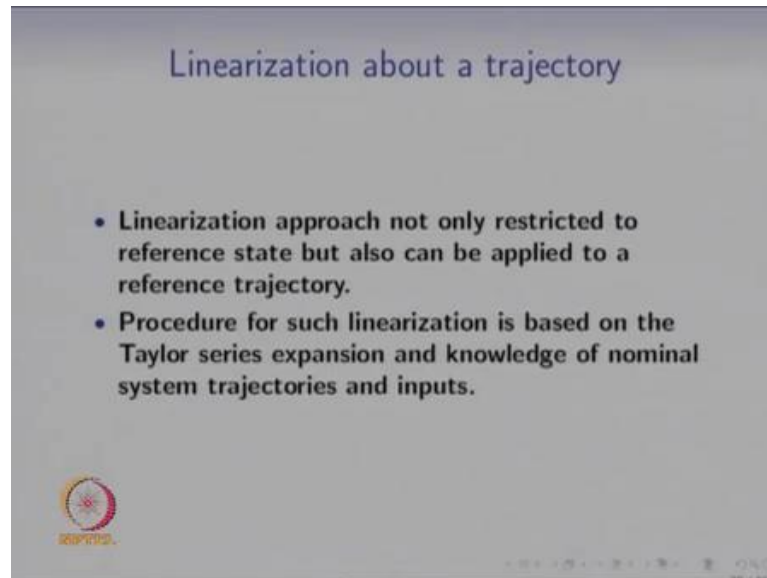


If, that is the case then we will very soon define that to be unstable optimal trajectory. Then, perhaps we can use feedback to stay close to this optimal trajectory, what is feedback here? Our control input was pre-decided? That pre-decided control input is indeed optimal, if all the modeling system modeling has been done to account for all uncertainties, but in the presence of uncertainties this actual input that we are giving is no longer optimal, because there are some model uncertainties, which we have not accounted for.

So, perhaps we could use feedback we could measure the actual sensor values now, and add the required value to the control input to the pre-decided control input. So, what is this linear controller supposed to do? It is supposed to design a linear controller to bring close by trajectories back to the optimal trajectory, if the trajectories that are close by are not coming back to the optimal trajectory. Then, we would like to design a controller that achieves this.

So, as far as this particular problem is concerned even, if the optimal input was pre-decided if the trajectory was not the optimal trajectory, then a linear controller suffices for stabilizing the system back to this optimal trajectory.

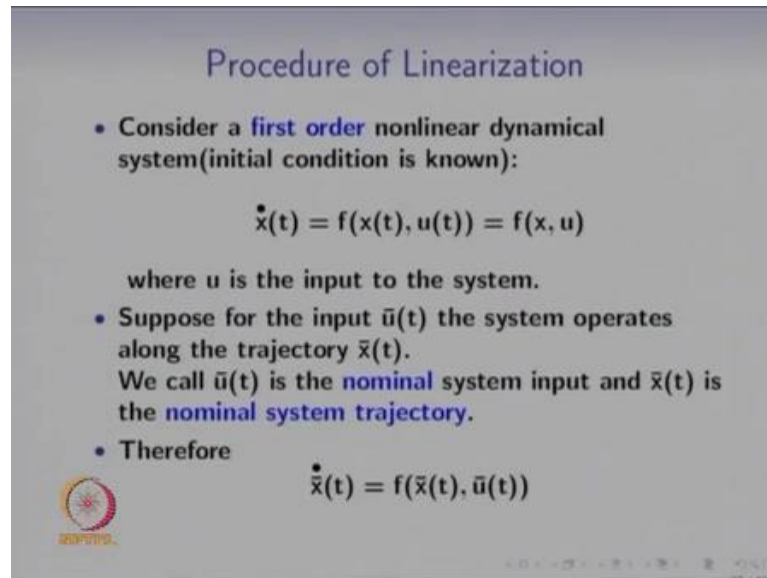
(Refer Slide Time: 33:14)



So, a linearization approach not only restricted to reference state not only restricted to the operating point, but it can also be applied to a reference trajectory to a operating trajectory. In the previous example, it was the optimal trajectory. This is the thing that we will study in detail. So, the procedure that we will see very soon is based on a Taylor series expansion and knowledge of the nominal system trajectories.

So, this reference state reference trajectory, we will call as nominal state value or nominal system trajectory the system trajectory itself has a corresponding nominal input. Also, what is the procedure now this is the next thing, we will see consider a first order non-linear dynamical system initial conditions are known \dot{x} is equal to f of x , u often. We suppress the t , we write \dot{x} is equal to f of x , u it is implicit that x and u are functions of time u is the input to the system. Suppose, for the input \bar{u} the system operates at along the trajectory \bar{x} .

(Refer Slide Time: 34:14)



The slide is titled "Procedure of Linearization" and contains the following content:

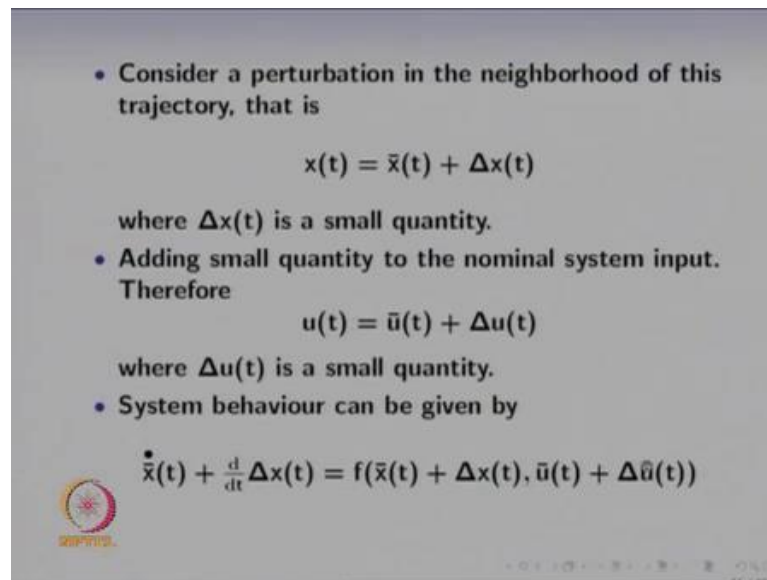
- Consider a **first order nonlinear dynamical system**(initial condition is known):
$$\dot{\bar{x}}(t) = f(x(t), u(t)) = f(x, u)$$
- where u is the input to the system.
- Suppose for the input $\bar{u}(t)$ the system operates along the trajectory $\bar{x}(t)$. We call $\bar{u}(t)$ is the **nominal system input** and $\bar{x}(t)$ is the **nominal system trajectory**.
- Therefore
$$\dot{\bar{x}}(t) = f(\bar{x}(t), \bar{u}(t))$$

There is a small circular logo in the bottom left corner of the slide.

So, this is our reference input and reference trajectory this is precisely the one that we were saying. This is perhaps the optimal trajectory, thus is the trajectory, x which turn out to be optimal in the sense that least fuel is consumed or the time taken is least. This is an example and for achieving this optimal trajectory. This is the input that we give, so when we say that this input is optimal, it is going to achieve this. Of course, this differential equation is satisfied this differential equation \dot{x} bar is equal to f of x bar comma u bar.

So, we will call u bar the nominal system input and x bar the nominal system trajectory. So, this nominal input turns out to exactly reproduce the nominal system trajectory, this is assumed why because u bar is pre-calculated it is like open loop control input. Now, we will consider a perturbation in the neighborhood of the trajectory. That is x t is equal to x bar plus some small perturbation δx . So, δx is a small quantity it is a small state value, we will also add a small quantity to the nominal system input.

(Refer Slide Time: 35:46)



- Consider a perturbation in the neighborhood of this trajectory, that is

$$x(t) = \bar{x}(t) + \Delta x(t)$$

where $\Delta x(t)$ is a small quantity.

- Adding small quantity to the nominal system input. Therefore

$$u(t) = \bar{u}(t) + \Delta u(t)$$

where $\Delta u(t)$ is a small quantity.

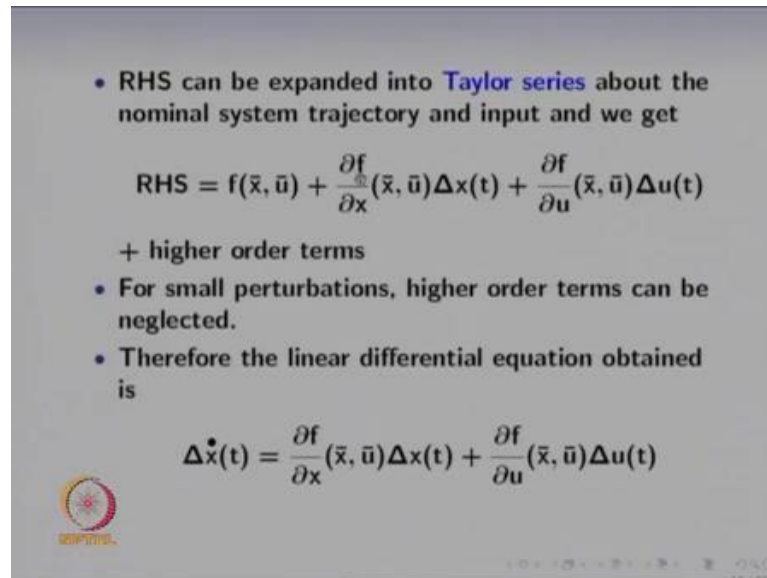
- System behaviour can be given by

$$\dot{\bar{x}}(t) + \frac{d}{dt} \Delta x(t) = f(\bar{x}(t) + \Delta x(t), \bar{u}(t) + \Delta \bar{u}(t))$$

So, u is equal to \bar{u} plus Δu , so please note this is the actual state, this is the nominal state, and this we consider as the deviation. Similarly, this is the nominal input this is the deviation and this gives the actual input. So, we are assuming that the deviations in the state and input are small. So, we can take derivative of this, so derivative of $x(t)$ is nothing but the derivative of this plus derivative of this. That is, what has been written on the left hand side here and on the right hand side.

We have to keep \bar{x} plus Δx and \bar{u} plus Δu because f is not a linear system, it is not a linear map. So, the right hand side right hand side of which equation right hand side of this equation. This is what we will call RHS right hand side. For the next slide the right hand side can be expanded into a Taylor series expansion about, what about the system trajectory.

(Refer Slide Time: 37:05)



- RHS can be expanded into **Taylor series** about the nominal system trajectory and input and we get

$$\text{RHS} = f(\bar{x}, \bar{u}) + \frac{\partial f}{\partial x}(\bar{x}, \bar{u})\Delta x(t) + \frac{\partial f}{\partial u}(\bar{x}, \bar{u})\Delta u(t)$$

+ higher order terms

- For small perturbations, higher order terms can be neglected.
- Therefore the linear differential equation obtained is

$$\Delta \dot{x}(t) = \frac{\partial f}{\partial x}(\bar{x}, \bar{u})\Delta x(t) + \frac{\partial f}{\partial u}(\bar{x}, \bar{u})\Delta u(t)$$

So, right hand side is equal to f of \bar{x} comma \bar{u} this is the 0 th order term in the Taylor series expansion plus derivative of f with respect to x times the deviation. This derivative of f with respect to x again depends on both x and u , we are going to evaluate it at \bar{x} comma \bar{u} why because our nominal system trajectory, and nominal system input are \bar{x} comma \bar{u} . We are seeking a Taylor series expansion about this nominal system trajectory and this nominal input. So, this is the 0 th order term and this is the first order term as far as Δx is concerned as far as x is concerned. We will also differentiate f with respect to u and again evaluate it at \bar{x} comma \bar{u} .

We will call this is the term corresponding to deviation in the input u plus. There are some higher order terms also these higher order terms involve second partial derivatives of f with respect to x and u . And they can be neglected under the assumptions that these deviations are small since we are asking the question whether the operating trajectory close whether trajectories close to the operating trajectory come back in that context. We are seeking an analysis of only about small deviations, and hence these higher order terms can be neglected.

So, after neglecting this we get this particular differential equation how is it that, we have got this differential equation directly from here. We have substituted in this differential equation the right hand side. We have substituted as what we will see in the next slide

and we have also used that \dot{x} bar equal to it is better that I write this on a slide the reference.

(Refer Slide Time: 38:54)

- Consider a perturbation in the neighborhood of this trajectory, that is

$$x(t) = \bar{x}(t) + \Delta x(t)$$

where $\Delta x(t)$ is a small quantity.

- Adding small quantity to the nominal system input. Therefore

$$u(t) = \bar{u}(t) + \Delta u(t)$$

where $\Delta u(t)$ is a small quantity.

- System behaviour can be given by

$$\dot{\bar{x}}(t) + \frac{d}{dt} \Delta x(t) = f(\bar{x}(t) + \Delta x(t), \bar{u}(t) + \Delta u(t))$$

(Refer Slide Time: 39:10)

$$\dot{\bar{x}} = f(\bar{x}, \bar{u})$$

Reference traj/input (Nominal)

$$x = \bar{x} + \Delta x, \quad u = \bar{u} + \Delta u$$

$$f(x, u) = f(\bar{x}, \bar{u}) + \left. \frac{\partial f}{\partial x} \right|_{(\bar{x}, \bar{u})} \Delta x + \left. \frac{\partial f}{\partial u} \right|_{(\bar{x}, \bar{u})} \Delta u + h.o.t.$$

Trajectory in the reference input satisfy the differential equation this is, what we also call the nominal trajectory and input. Then, we are going to analyze not at \bar{x} and \bar{u} , but x is equal to \bar{x} plus Δx . And u is equal to \bar{u} plus Δu . It is about this particular x and u that, we are going to analyze, and for that purpose, we wrote f of x

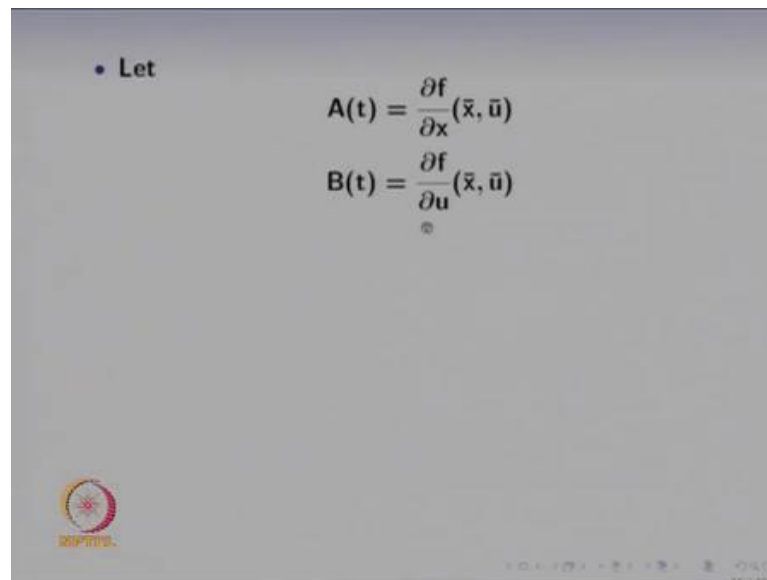
comma u is equal to f of x bar comma u bar plus del f by del x evaluated at x bar comma u bar times delta x plus del f by del u times delta u times delta u. This again evaluated at x bar comma u bar.

(Refer Slide Time: 40:54)

$$\frac{d}{dt} x = \frac{d}{dt} \bar{x} + \frac{d}{dt} \Delta x \approx f(\bar{x}, \bar{u}) + \left. \frac{\partial f}{\partial x} \right|_{(\bar{x}, \bar{u})} \Delta x + \left. \frac{\partial f}{\partial u} \right|_{(\bar{x}, \bar{u})} \Delta u$$

So, this we have expanded f of x comma u as 0 th order term, and together as first order term plus higher order terms, which we have decided to neglect because the deviations x and u are small. Now, we are going to substitute this back into this differential equation into which differential equation at least the derivative operator is a linear operator because of which we get this. That is equal to after the neglecting of higher order terms in which this and this both have been evaluated at x bar comma u bar, strictly speaking this is only approximately equal to why because we have decided to ignore the higher order terms.

(Refer Slide Time: 42:06)



• Let

$$A(t) = \frac{\partial f}{\partial \bar{x}}(\bar{x}, \bar{u})$$
$$B(t) = \frac{\partial f}{\partial \bar{u}}(\bar{x}, \bar{u})$$

So, we will now define this particular term this particular constant matrix as a, and this particular matrix as b. This is the most important thing that we do, when we linearize the system. This is what, we will continue in the next lecture. So, we just see the definitions here before we stop for today's lecture. So, we are going to now define a as this particular matrix and b as this matrix this is, what we will see in the following lecture with that we end lecture number 12. In the next lecture, we will see more about the matrices a and b and why they are time varying.

Thank you.