

Scilab Textbook Companion for  
Nonlinear Dynamics And Chaos  
by S. H. Strogatz<sup>1</sup>

Created by  
Ankur Agarwal  
B.Tech (pursuing)  
Electronics Engineering  
LNMIIT, Jaipur  
College Teacher  
Manish Dev Shrimali, LNMIIT  
Cross-Checked by  
Santosh Kumar, IITB

July 31, 2019

<sup>1</sup>Funded by a grant from the National Mission on Education through ICT, <http://spoken-tutorial.org/NMEICT-Intro>. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website <http://scilab.in>

# Book Description

**Title:** Nonlinear Dynamics And Chaos

**Author:** S. H. Strogatz

**Publisher:** Levant Books (Indian Publisher)

**Edition:** 1

**Year:** 2007

**ISBN:** 81-87169-85-0

Scilab numbering policy used in this document and the relation to the above book.

**Exa** Example (Solved example)

**Eqn** Equation (Particular equation of the above book)

**AP** Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

# Contents

List of Scilab Codes	4
2 Flows on the Line	6
3 Bifurcations	23
4 Flows on the Circle	39
5 Linear Systems	51
6 Phase Plane	75
7 Limit Cycles	91
8 Bifurcations Revisited	112
10 One Dimensional Maps	129
11 Fractals	142

# List of Scilab Codes

Exa 2.2.1	Fixed Points and Stability . . . . .	6
Exa 2.2.2	Fixed Points and Stability . . . . .	8
Exa 2.2.3	Fixed Points and Stability . . . . .	12
Exa 2.4.1	Linear Stability Analysis . . . . .	13
Exa 2.4.2	Linear Stability Analysis . . . . .	15
Exa 2.4.3	Linear Stability Analysis . . . . .	18
Exa 2.7.1	Potentials . . . . .	20
Exa 2.7.2	Potentials . . . . .	21
Exa 3.1.1	Saddle Node Bifurcation . . . . .	23
Exa 3.1.2	Saddle Node Bifurcation . . . . .	26
Exa 3.2.1	Transcritical Bifurcation . . . . .	28
Exa 3.2.2	Transcritical Bifurcation . . . . .	31
Exa 3.4.1	Pitchfork Bifurcation . . . . .	35
Exa 3.4.2	Pitchfork Bifurcation . . . . .	37
Exa 4.1.1	Examples and Definitions . . . . .	39
Exa 4.1.2	Examples and Definitions . . . . .	44
Exa 4.3.1	Nonuniform Oscillator . . . . .	44
Exa 4.6.1	Superconducting Josephson Junctions . . . . .	49
Exa 5.1.1	Definitions and Examples . . . . .	51
Exa 5.1.2	Definitions and Examples . . . . .	55
Exa 5.2.1	Classification of Linear Systems . . . . .	64
Exa 5.2.2	Classification of Linear Systems . . . . .	64
Exa 5.2.3	Classification of Linear Systems . . . . .	68
Exa 5.2.4	Classification of Linear Systems . . . . .	69
Exa 5.2.6	Classification of Linear Systems . . . . .	73
Exa 5.2.7	Classification of Linear Systems . . . . .	73
Exa 6.1.1	Phase Portraits . . . . .	75
Exa 6.3.1	Fixed Points and Linearization . . . . .	78

Exa 6.3.2	Fixed Points and Linearization . . . . .	81
Exa 6.5.2	Conservative Systems . . . . .	84
Exa 6.6.1	Reversible Systems . . . . .	88
Exa 6.6.3	Reversible Systems . . . . .	89
Exa 7.1.1	A Simple Limit Cycle . . . . .	91
Exa 7.1.2	Van der Pol Oscillator . . . . .	95
Exa 7.2.3	Ruling Out Closed Orbits . . . . .	96
Exa 7.2.5	Ruling Out Closed Orbits . . . . .	98
Exa 7.3.1	Poincare Bendixson Theorem . . . . .	100
Exa 7.3.2	Poincare Bendixson Theorem . . . . .	102
Exa 7.3.3	Poincare Bendixson Theorem . . . . .	106
Exa 7.4.1	Lienard Systems . . . . .	107
Exa 7.5.1	Relaxation Oscillations . . . . .	109
Exa 8.1.1	Saddle Node Transcritical Pitchfork Bifurca- tions . . . . .	112
Exa 8.1.2	Saddle Node Transcritical Pitchfork Bifurca- tions . . . . .	115
Exa 8.1.3	Saddle Node Transcritical Pitchfork Bifurca- tions . . . . .	119
Exa 8.2.1	Hopf Bifurcations . . . . .	121
Exa 8.3.1	Oscillating Chemical Reactions . . . . .	123
Exa 8.3.2	Oscillating Chemical Reactions . . . . .	125
Exa 10.1.1	Fixed Points and Cobwebs . . . . .	129
Exa 10.1.2	Fixed Points and Cobwebs . . . . .	130
Exa 10.3.1	Logistic Map . . . . .	133
Exa 10.3.2	Logistic Map . . . . .	137
Exa 10.3.3	Logistic Map . . . . .	137
Exa 10.6.1	Universality and Experiments . . . . .	138
Exa 10.7.2	Renormalization . . . . .	140
Exa 10.7.3	Renormalization . . . . .	141
Exa 11.3.1	Dimension of Self Similar Fractals . . . . .	142
Exa 11.3.2	Dimension of Self Similar Fractals . . . . .	142
Exa 11.3.3	Dimension of Self Similar Fractals . . . . .	143
Exa 11.4.1	Box Dimension . . . . .	144
Exa 11.4.2	Box Dimension . . . . .	144

# List of Figures

2.1	Fixed Points and Stability	7
2.2	Fixed Points and Stability	9
2.3	Fixed Points and Stability	10
2.4	Fixed Points and Stability	12
2.5	Linear Stability Analysis	14
2.6	Linear Stability Analysis	15
2.7	Linear Stability Analysis	17
2.8	Potentials	19
2.9	Potentials	21
3.1	Saddle Node Bifurcation	24
3.2	Saddle Node Bifurcation	27
3.3	Transcritical Bifurcation	29
3.4	Transcritical Bifurcation	32
3.5	Pitchfork Bifurcation	34
3.6	Pitchfork Bifurcation	35
3.7	Pitchfork Bifurcation	37
4.1	Examples and Definitions	40
4.2	Examples and Definitions	41
4.3	Nonuniform Oscillator	45
4.4	Nonuniform Oscillator	46
4.5	Superconducting Josephson Junctions	49

5.1	Definitions and Examples	52
5.2	Definitions and Examples	53
5.3	Definitions and Examples	56
5.4	Classification of Linear Systems	65
5.5	Classification of Linear Systems	66
5.6	Classification of Linear Systems	67
5.7	Classification of Linear Systems	70
5.8	Classification of Linear Systems	71
6.1	Phase Portraits	76
6.2	Fixed Points and Linearization	79
6.3	Fixed Points and Linearization	81
6.4	Fixed Points and Linearization	82
6.5	Conservative Systems	85
6.6	Conservative Systems	86
7.1	A Simple Limit Cycle	92
7.2	A Simple Limit Cycle	93
7.3	Van der Pol Oscillator	95
7.4	Ruling Out Closed Orbits	97
7.5	Ruling Out Closed Orbits	99
7.6	Poincare Bendixson Theorem	101
7.7	Poincare Bendixson Theorem	103
7.8	Poincare Bendixson Theorem	104
7.9	Poincare Bendixson Theorem	105
7.10	Lienard Systems	108
7.11	Relaxation Oscillations	110
8.1	Saddle Node Transcritical Pitchfork Bifurcations	113
8.2	Saddle Node Transcritical Pitchfork Bifurcations	114
8.3	Saddle Node Transcritical Pitchfork Bifurcations	116
8.4	Saddle Node Transcritical Pitchfork Bifurcations	117
8.5	Saddle Node Transcritical Pitchfork Bifurcations	120
8.6	Hopf Bifurcations	122
8.7	Oscillating Chemical Reactions	124
8.8	Oscillating Chemical Reactions	126
8.9	Oscillating Chemical Reactions	127
10.1	Fixed Points and Cobwebs	131



10.2 Fixed Points and Cobwebs . . . . .	131
10.3 Logistic Map . . . . .	134
10.4 Logistic Map . . . . .	136
10.5 Universality and Experiments . . . . .	139

# Chapter 2

## Flows on the Line

Scilab code Exa 2.2.1 Fixed Points and Stability

```
1 //Example 2.2.1 Page 19
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","on") //hold off
9
10 x=poly(0,"x");
11 f = (x^2)-1; //Defining
   Polynomial—> x(dot)=x^2 -1. Let this be f(x)
12 disp("Fixed Points are :")
13 y = roots(f)
14
15 set(gca(),"auto_clear","off") //hold on
16 set(gca(),"grid",[2,5])
17 for (x=-5:0.1:5)
```

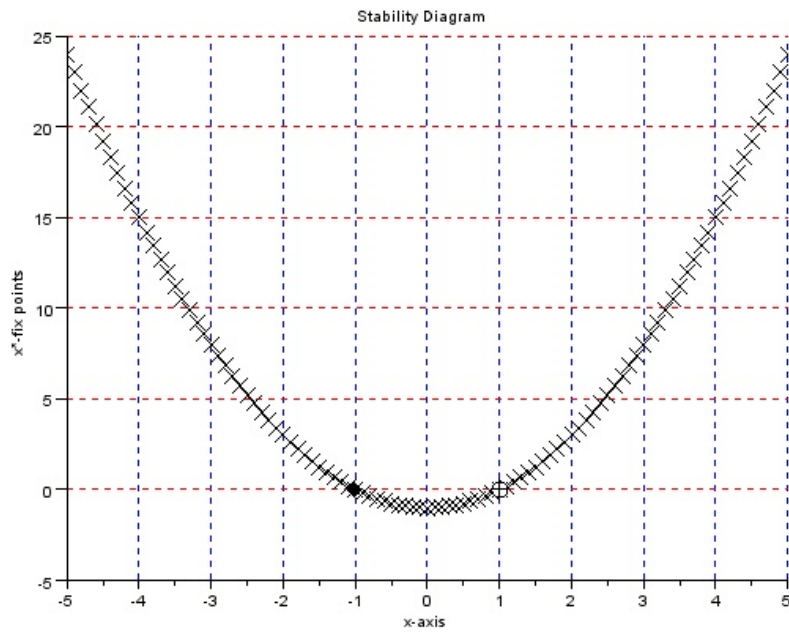


Figure 2.1: Fixed Points and Stability

```

18     plot2d(x,(x^2)-1,style=-2)
19
20 end
21     plot2d(-1,0,style=-4)           //To show "Diamond
    Sign" --> Representing a Fixed Stable Point.
22     plot2d(1,0,style=-3)           //To Show "a Cross
    within Circle" --> Representing a Fixed
    Unstable Point.
23
24     xtitle("Stability Diagram","x-axis","x*-fix
    points")
25
26 disp("From the diagram it is clear that x=-1 is the
    Fixed Stable Point.")
27 disp("And x=+1 is the Unstable Fixed Point.")
28
29
30 //End of Example 2.2.1

```

---

### Scilab code Exa 2.2.2 Fixed Points and Stability

```

1 //Example 2.2.2 Page 20
2 //Non-Linear Dynamics and Chaos, First Indian
    Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","on") //hold off
9

```

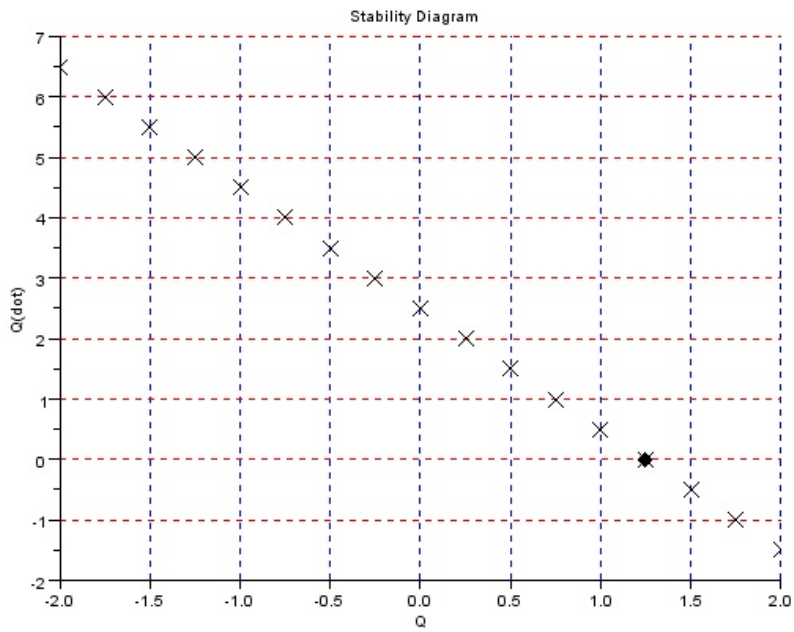


Figure 2.2: Fixed Points and Stability

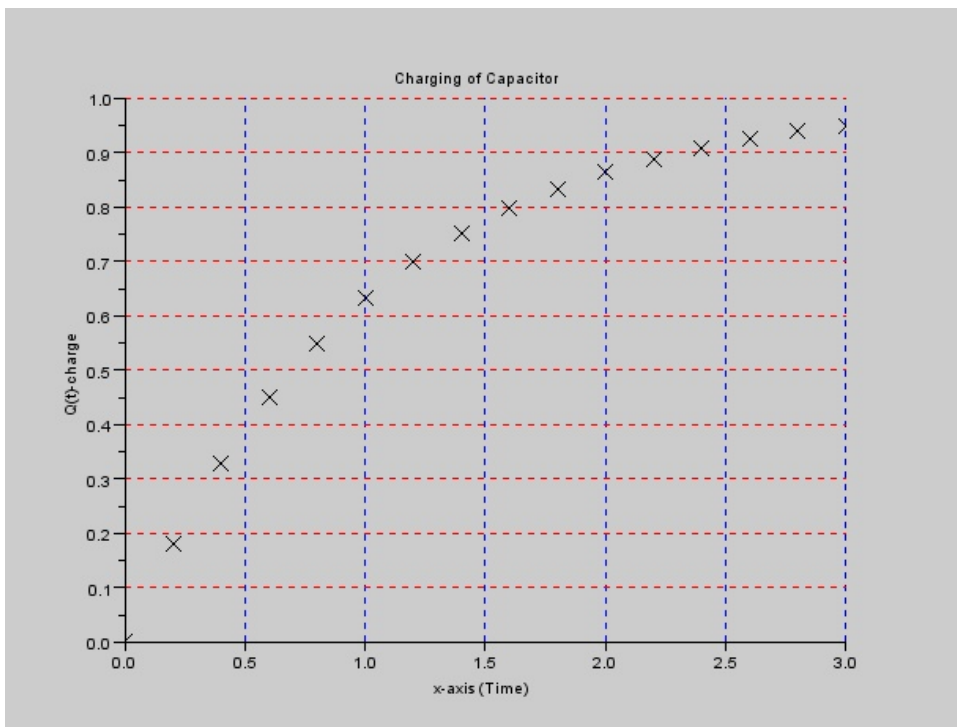


Figure 2.3: Fixed Points and Stability

```

10 //As Resistance(R) , Capacitance(C) and Voltage(V(0))
    are constants so
11 //Let R=4 , V(0)=10 , RC = 0.5 (Thus, C = 0.125) to
    plot Q(dot) v/s Q.
12 // Thus, V(0)/R = 2.5
13 //Q(dot) = f(Q) = 2.5 - Q/0.5 => Q(dot) = f(Q) = 2.5
    - 2*Q.
14
15 set(gca(),"auto_clear","off") //hold on
16 for Q = -2:0.25:2
17     y = 2.5 - 2*Q ; //y=Q(dot)=f(Q)
18     plot2d(Q,y,style=-2)
19 end
20 plot2d(1.25,0,style=-4) //Just to depict
    Stable Fixed Point
21
22 xtitle("Stability Diagram","Q","Q(dot)")
23 set(gca(),"grid",[2,5]) //Grid on
24
25 disp("From the Figure of Q(dot) v/s Q => we can find
    the Fixed Point (where Q(dot) crosses X-Axis)")
26
27 disp("Put Q(dot)=0 => Q* = CV(0), Q* = CV(0) = 1.25
    in our case.")
28 disp("From the figure itself it is clear that the
    Fixed Point is Stable.")
29
30
31 // Now Second Part of the question.
32 //From our knowledge of Electronics and Twelfth Class
    we know that :-
33 //Charging of Capacitor is described as
    Exponentially increasing with time. (-exp(-t))
34 figure //To get a new Graphic Window
35
36 for t=0:0.2:3
37     Q = 1-exp(-t); //To show exponential
        increasing nature of graph starting from Q=0.

```

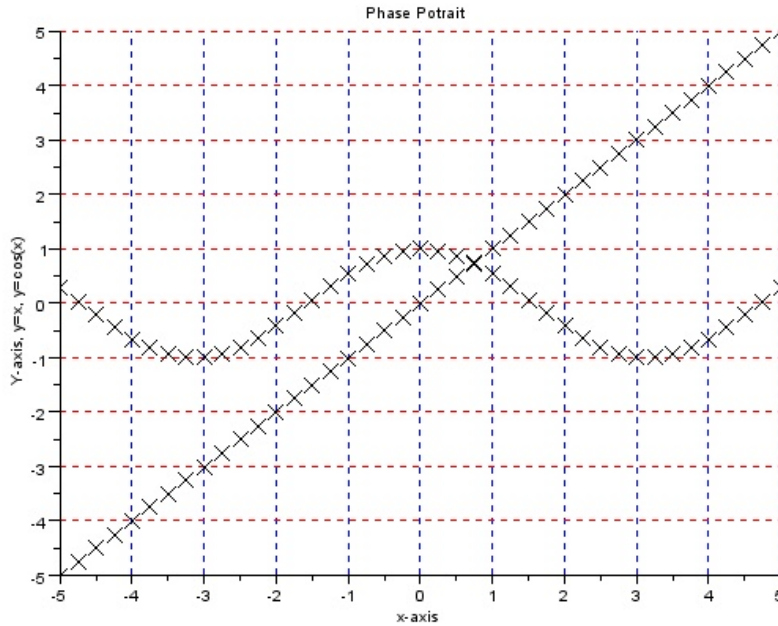


Figure 2.4: Fixed Points and Stability

```

38     plot2d(t,Q,style=-2)
39 end
40 set(gca(),"grid",[2,5])
41 xtitle("Charging of Capacitor","x-axis (Time)","Q(t)
    -charge")
42 //End of Example 2.2.2

```

---

### Scilab code Exa 2.2.3 Fixed Points and Stability

```

1 //Example 2.2.3 Page 21
2 //Non-Linear Dynamics and Chaos, First Indian

```



```

    Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off") //hold on
9
10 for x=-5:0.25:5
11     y1 = x;
12     y2 = cos(x);
13     plot2d(x,y1,style=-2)
14     plot2d(x,y2,style=-2)
15 end
16 //plot2d(0.75,0.75, style=-3) //To show that
    the fixed point is Unstable.
17 set(gca(),"grid",[2,5])
18 xtitle("Phase Potrait","x-axis","Y-axis , y=x, y=cos(
    x)")
19
20 disp ("OBSERVATIONS from the Graph :-")
21 disp("1. y=x and y=cos(x) intersects at exactly one
    point.")
22 disp("2. The fixed point is Unstable ")
23
24
25 //End of Example 2.2.3 Page 21.

```

---

#### Scilab code Exa 2.4.1 Linear Stability Analysis

```

1 //Example 2.4.1 Page 25
2 //Non-Linear Dynamics and Chaos, First Indian
    Edition Print 2007

```

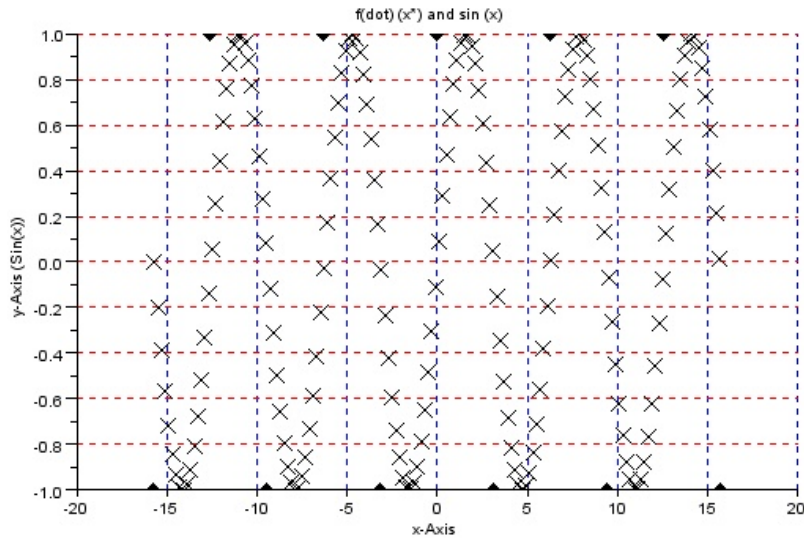


Figure 2.5: Linear Stability Analysis

```

3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off") //hold on
9
10
11 disp("When x(double dot) < 0 then => Fixed Point is
12     Stable.")
13
14 disp("When x(double dot) > 0 then => Fixed Point is
15     Unstable.")
16
17 //x(dot) = f(x) = sin(x) = 0 => x* = k*pi
18
19 for k= -5*(%pi):%pi:5*(%pi)
20     f1 = cos(k)
21     plot2d(k,f1,style=-4)
22 end

```

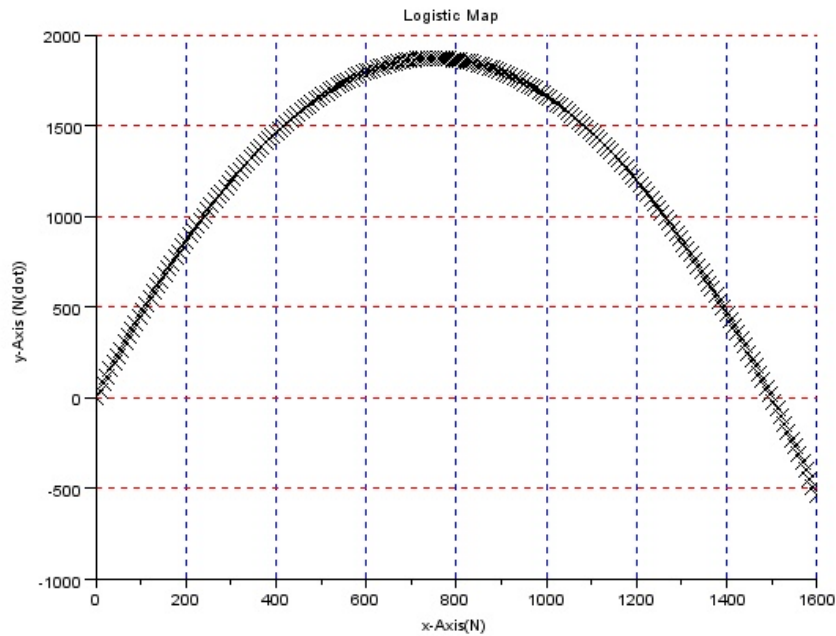


Figure 2.6: Linear Stability Analysis

```

20
21 for x= -5*(%pi):0.2:5*(%pi)
22     f = sin(x);
23     plot2d(x,f,style=-2)
24 end
25 set(gca(),"grid",[2,5])
26 xtitle("f(dot) (x*) and sin (x)","x-Axis","y-Axis (
    Sin(x)")

```

---

Scilab code Exa 2.4.2 Linear Stability Analysis

```

1 //Example 2.4.2 Page 25
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off")           //hold on
9
10 //f(N) = N(dot) = r*N(1-N/K)
11 // Let r = 5 and K = 1600; to plot N(dot) v/s N
12
13 for N = 0:8:1600
14     f = 5*N*(1-(N/1500));
15     plot2d(N,f,style=-2)
16 end
17 set(gca(),"grid",[2,5])
18 xtitle("Logistic Map","x-Axis(N)","y-Axis (N(dot))")
19
20 disp("From graph of N(dot) v/s N it is obvious that
   :-")
21 disp("N = 0 and N=K (1500 in our case) are fixed
   points.")
22 disp("And N=0 is Unstable Fixed Point and N=K is
   Stable Fixed Point.")
23
24 disp("When N(double dot) < 0 then => Fixed Point is
   Stable.")
25 disp("When N(double dot) > 0 then => Fixed Point is
   Unstable.")
26
27 //End of Example 2.4.2

```

---

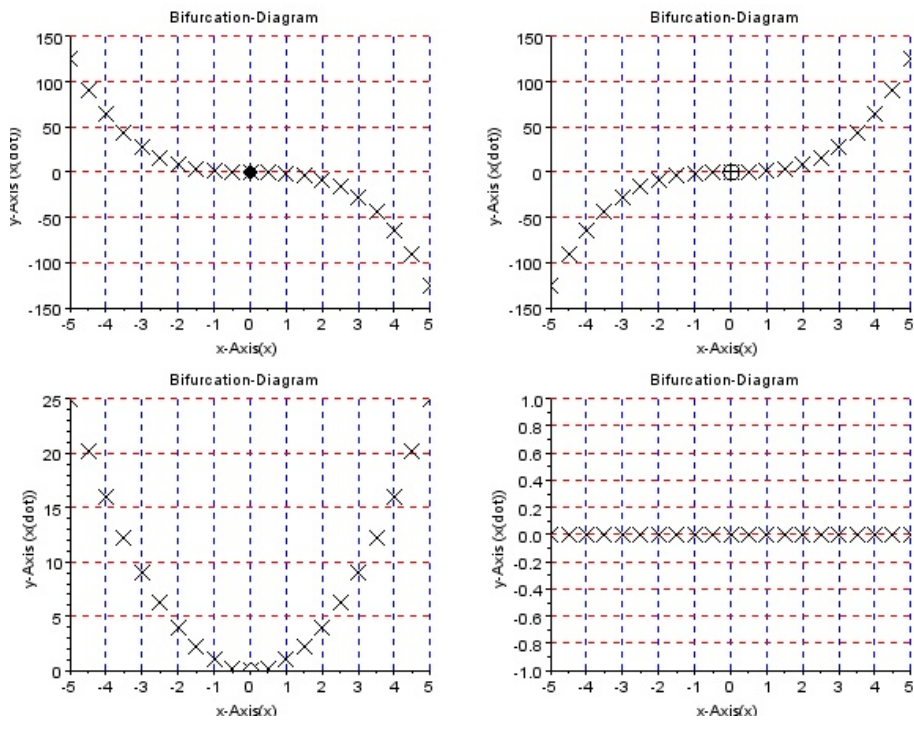


Figure 2.7: Linear Stability Analysis

### Scilab code Exa 2.4.3 Linear Stability Analysis

```
1 //Example 2.4.3 Page 25
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca()," auto_clear"," off")           //hold on
9
10
11 for x = -5:0.5:5
12     f1 = -(x^3);
13     subplot(221)
14     plot2d(x,f1,style=-2)
15     plot2d(0,0,style=-4)
16     set(gca()," grid",[2,5])
17     xtitle(" Bifurcation -Diagram"," x-Axis(x)"," y-Axis
        (x(dot))")
18
19     f2 = x^3;
20     subplot(222)
21     plot2d(x,f2,style=-2)
22     plot2d(0,0,style=-3)
23     set(gca()," grid",[2,5])
24     xtitle(" Bifurcation -Diagram"," x-Axis(x)"," y-Axis
        (x(dot))")
25
26     f3 = x^2;
27     subplot(223)
28     plot2d(x,f3,style=-2)
29     plot2d(0,0,style=-5)
30     set(gca()," grid",[2,5])
31     xtitle(" Bifurcation -Diagram"," x-Axis(x)"," y-Axis
        (x(dot))")
32
```

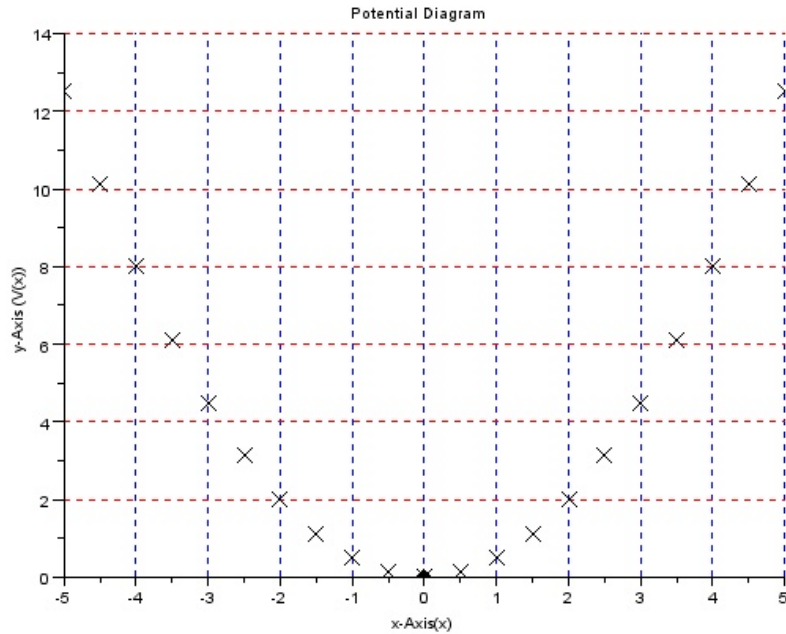


Figure 2.8: Potentials

```

33     f4 = 0;
34     subplot(224)
35     plot2d(x,f4,style=-2)
36     set(gca(),"grid",[2,5])
37     xtitle("Bifurcation -Diagram", "x-Axis(x)", "y-Axis
           (x(dot))")
38
39 end
40 //set(gca(),"grid",[2,5])
41 //xtitle("Logistic Map", "x-Axis(N)", "y-Axis (N(dot))
           ")

```

---

### Scilab code Exa 2.7.1 Potentials

```
1 //Example 2.7.1 Page 31
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off")           //hold on
9
10 //x(dot) = f(x) = -x
11 // On Integrating we get V(x) = (1/2)x^2 + C ; C =
   0.
12
13 //Now, Plotting the potential i.e. V(x)
14
15 for x = -5:0.5:5
16     V = (1/2)*x^2;
17     plot2d(x,V,style=-2)
18 end
19 plot2d(0,0,style=-4) //Just to show that the
   fixed point is Stable.
20 set(gca(),"grid",[2,5])
21 xtitle("Potential Diagram","x-Axis(x)","y-Axis (V(x)
   )")
22
23 disp("From graph of V(x) v/s x itself , it is clear
   that :-")
24 disp("The only equilibrium point is at x=0")
25 disp(" And from the flow we can conclude that the x
   *=0 point is Stable.")
26
27 disp("OR")
28 disp("f(dot)(x) = -1 for all x; Thus every
   equilibrium point which exists is Stable.")
29
```



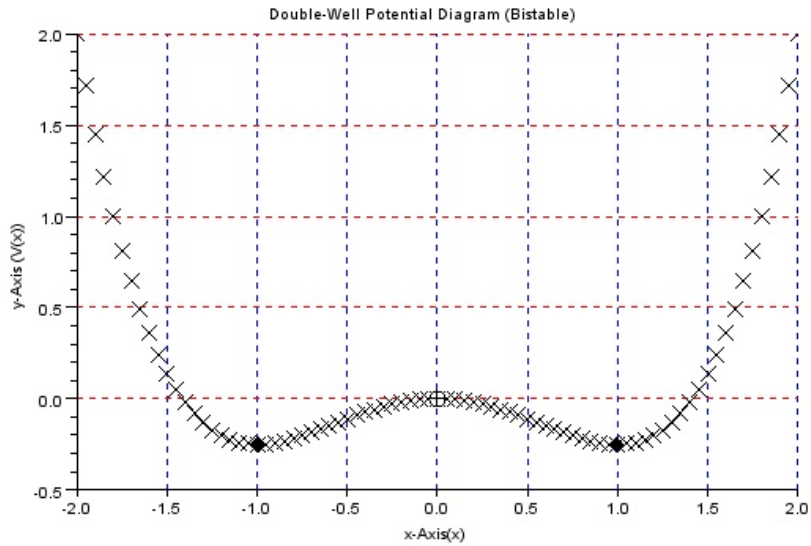


Figure 2.9: Potentials

30 //End of Example.

---

#### Scilab code Exa 2.7.2 Potentials

```

1 //Example 2.7.2 Page 31
2 //Non-Linear Dynamics and Chaos, First Indian
  Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off") //hold on
9

```

```

10 //Given : x(dot) = f(x) = x - x^3.
11 //On integrating we get --> V(x) = -(1/2)x^2 +
    (1/4)x^4 + C ; C=0.
12
13 //Now plotting V(x) v/s x ; and observe fix points
    and their Stabilities
14
15 for x = -2:0.05:2
16     V = -(1/2)*(x^2) + (1/4)*(x^4);
17     plot2d(x,V,style=-2)
18 end
19 plot2d(0,0,style=-3) //Just to show that the
    fixed point is UnStable.
20 plot2d(-1,-1/4,style=-4) //Just to show that the
    fixed point is Stable.
21 plot2d(1,-1/4,style=-4) //Just to show that the
    fixed point is Stable.
22 set(gca(),"grid",[2,5])
23 xtitle("Double-Well Potential Diagram (Bistable)","x
    -Axis(x)","y-Axis (V(x))")

```

---

# Chapter 3

## Bifurcations

Scilab code Exa 3.1.1 Saddle Node Bifurcation

```
1 // Example 3.1.1, Page = 47
2 // Non-Linear Dynamics and Chaos, First Indian
  Edition
3 //Steven H. Strogatz
4 //  $x(\text{dot})=f(x)=r-(x^2)$ 
5 // Notation:  $x_1, x_2, x_3 \dots$  are the fixed point
  solutions
6 // since  $x*$  is an error, because of multiplication
  operator.
7
8 clear;
9 clc;
10 close;
11
12 for r=-1:0.1:+1 //Varying value of
  parameter "r" so as to obtain bifurcation diagram
  .
13
14     x1 = +(sqrt(r)); //First Fixed Point.
```

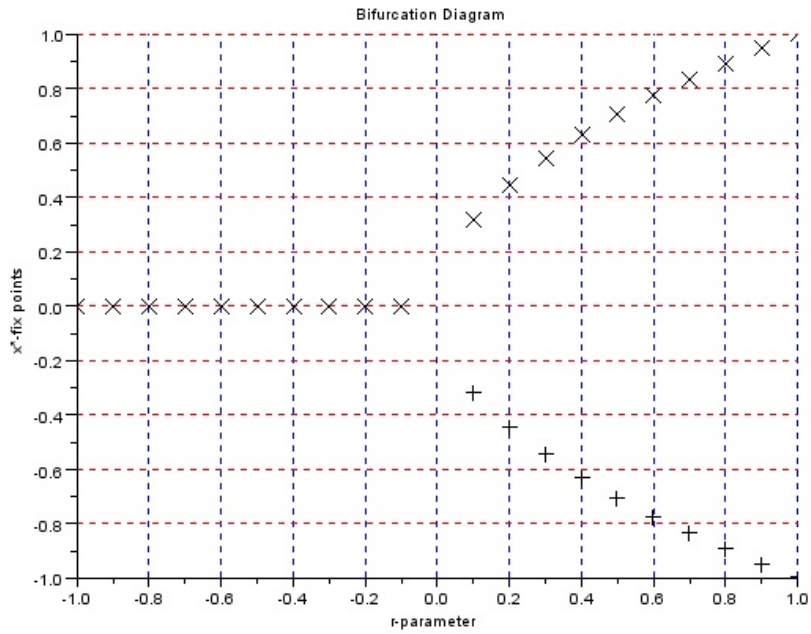


Figure 3.1: Saddle Node Bifurcation

```

15     x2 = -(sqrt(r));           //Second Fixed Point.
16     f1 = -2*x1;              //f'(x) at x1.
17     f2 = -2*x2;              //f'(x) at x2.
18
19     //x(double dot) = f'(x) = -2*x.
20     set(gca(),"auto_clear","off") //hold on
21     set(gca(),"grid",[2,5])
22     if (r<0) then
23         y = 0; //just to draw y=0 line for r<0,
                as no solution
24         disp(r)
25         disp("No fixed points.")
26         plot2d(r,y,style=-2)
27
28     else
29         set(gca(),"auto_clear","off")
30         if(f1>0) //unstable solutions
31             disp(r,x1)
32             disp("unstable solutions")
33             plot2d(r,x1,style=-1)
34
35         end
36         if(f1<0) //Stable solutions
37             disp(r,x1)
38             disp("stable solutions")
39             plot2d(r,x1,style=-2)
40
41         end
42
43         set(gca(),"auto_clear","off")
44
45         if(f2>0) //unstable solutions
46             disp(r,x2)
47             disp("unstable solutions")
48             plot2d(r,x2,style=-1)
49
50         end
51         if(f2<0) //stable solutions

```

```

52         disp(r,x2)
53         disp("stable solutions")
54         plot2d(r,x2,style=-2)
55
56     end
57     xtitle("Bifurcation Diagram","r-parameter","
           x*-fix points")
58 end
59 end
60 disp("Clearly from the graph x=0 is bifurcation
       point.")
61 set(gca(),"auto_clear","on") //hold off
62 //End of Example

```

---

### Scilab code Exa 3.1.2 Saddle Node Bifurcation

```

1 // Example 3.1.2 Pg 47
2 //Non-Linear Dynamics and Chaos, First Indian
  Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 mtlb_hold off
9
10 for(r=0:1:3) //Varying value of
    parameter "r" to see number of fixed point
    solutions.
11     x=-2:0.1:3;
12     set(gca(),"grid",[2,5]);
13     mtlb_hold on
14     plot2d(x,exp(-x),style=-4);

```

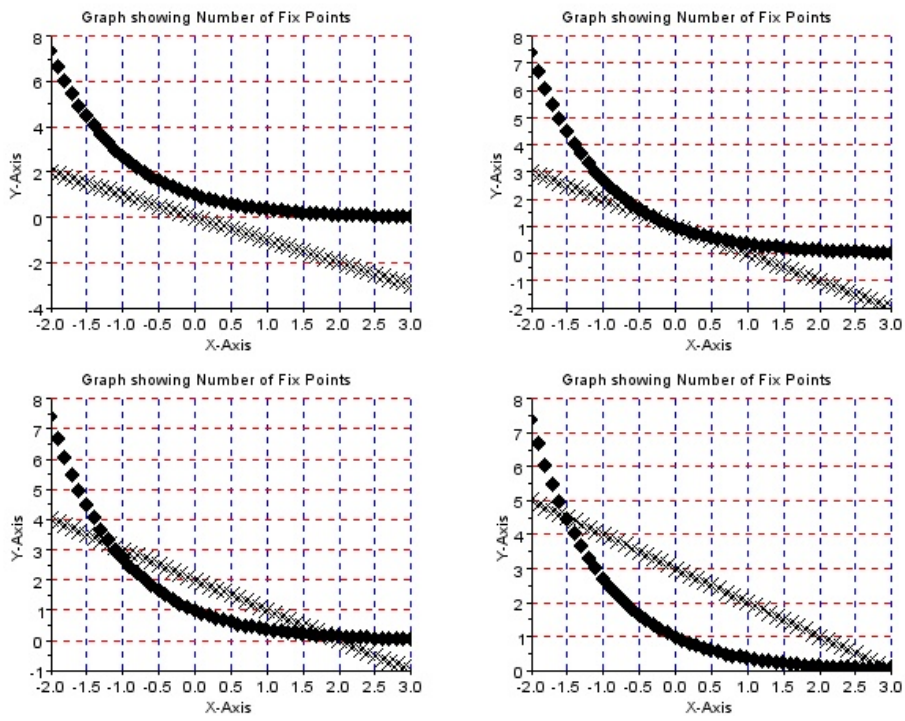


Figure 3.2: Saddle Node Bifurcation

```

15     plot2d(x,r-x,style=-2);
16     figure;           //to get new graphics window
17     set(gca(),"grid",[2,5])
18     xtitle("Graph showing Number of Fix Points","X-
           Axis","Y-Axis");
19 end
20
21     disp("From the graph we get intersection point")
22     disp("And hence we got our FIXED POINT SOLUTION.
           ")
23     disp("Clearly from graph we get stable solution
           when line is below exp(-x) graph.")
24     disp("Unstable solution when line is above exp(-
           x) graph.")
25     disp("From graph we infer that :")
26     disp("1. No Fixed Points for r<1")
27     disp("2. One Fixed Point when r=1.")
28     disp("3. Two Fixed Points for r>1.")
29     disp("hence Bifurcation Point is clearly , r(c)=1
           ")
30 mtlb_hold off
31 //End of Example

```

---

### Scilab code Exa 3.2.1 Transcritical Bifurcation

```

1 //Example 3.2.1 Page 51
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition, Print-2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;

```



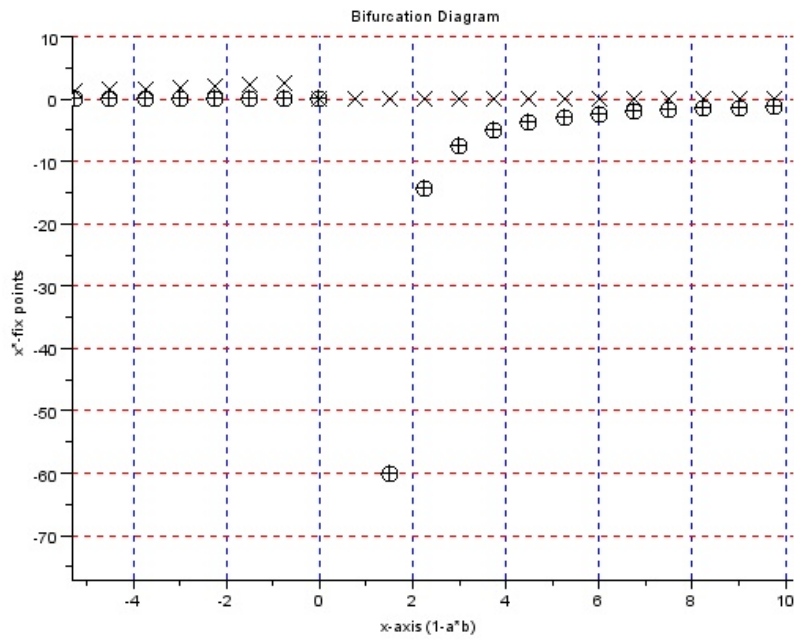


Figure 3.3: Transcritical Bifurcation

```

8
9 // General INTRODUCTION
10 disp("To show their is Transcritical Bifurcation
      show :")
11 disp("1. Their are always two fixed points ,")
12 disp("And they change their Stability around
      Bifurcation Point.")
13 disp("2. Hence, the nature of given equation should
      be Quadratic.")
14 disp("To show this use Taylor Expansion.")
15 // END INTRODUCTION
16
17 //By Taylor Expansion of given x(dot) and neglecting
      order(x^3) and higher order terms we get ——
18 //x(dot) = (1-ab)x + (1/2)(ab*b)x^2.-----Let x(dot)
      = f(x)
19
20 //Stability Analysis : Put x(dot)=f(x)=0
21 for(a=5) //Fixing one
      of the two parameters (i.e "a" And "b")
22     for(b=-4:0.15:3) //Varying
          another parameter to plot Bifurcation Diagram
23
24         x1 = 0; //First Fixed Point
25         x2 = 2*(a*b -1)/(a*b*b); //Second Fixed
          Point
26         f1 = (1-a*b); //f'(x) at x1
27         f2 = -(1-a*b); //f'(x) at x2
28
29         //x(double dot)= f'(x) = (1-ab) + a(b^2)x
30
31         set(gca(),"auto_clear","off") //For "
          hold on" to plot points.
32         set(gca(),"grid",[2,5]) // To
          set axis
33         if(f1>0) //Unstable fixed
          point.

```

```

34         plot2d(1-a*b,x1,style=-2)
35     end
36     if(f1<0)                                     //Stable Fixed point
37         .
38         plot2d(1-a*b,x1,style=-3)
39     end
40     if(f2>0)                                     //Unstable Fixed
41         Point.
42         plot2d(1-a*b,x2,style=-2)
43     end
44     if(f2<0)                                     //Stable Fixed Point
45         .
46         plot2d(1-a*b,x2,style=-3)
47     end
48     xtitle(" Bifurcation Diagram", "x-axis (1-a*b)
49         ", "x*-fix points")
50 end
51 disp("From Graph it is clearly visible that the two
52     fixed points changes stability around point ab=1
53     ")
54
55 set(gca()," auto_clear", "on")                 //for hold off
56
57 //end of Example

```

---

### Scilab code Exa 3.2.2 Transcritical Bifurcation

```

1 //Example 3.2.2 Page 52
2 //Non-Linear Dynamics and Chaos, First Indian
3   Edition, Print-2007
4 //Steven H. Strogatz
5

```

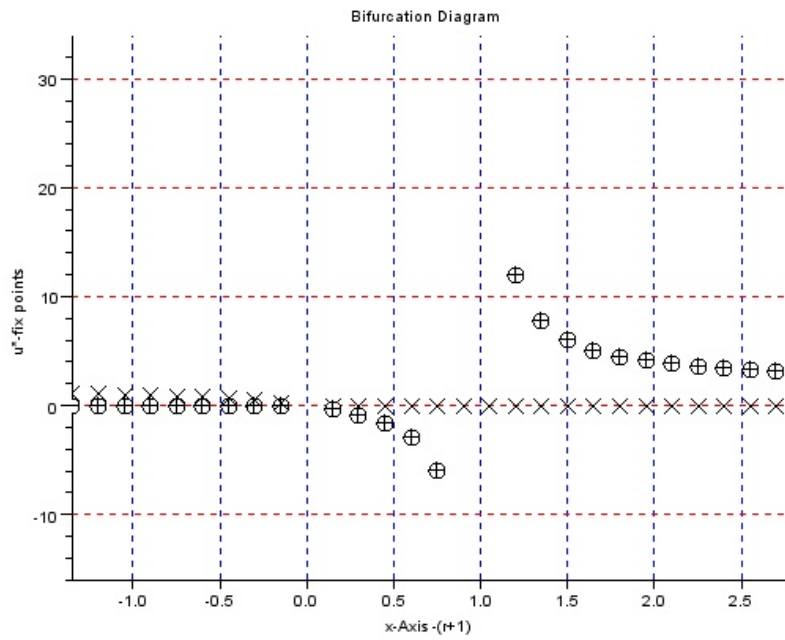


Figure 3.4: Transcritical Bifurcation

```

5 clear;
6 clc;
7 close;
8
9 //General INTRODUCTION
10 disp("To show their is Transcritical Bifurcation
      show :")
11 disp("1. Their are always two fixed points ,")
12 disp("And they change their Stability around
      Bifurcation Point.")
13 disp("2. Hence, the nature of given equation should
      be Quadratic.")
14 disp("To show this use Taylor Expansion.")
15 // End INTRODUCTION
16
17 // u(dot) = (r+1)u - (1/2)r(u^2) + O(u^3)
18 // u(dot) = (r+1)u - (1/2)r(u^2) + ZERO -----
      neglecting higher order terms
19 // Let u(dot) = f(u)
20
21 for r=-4:0.15:3           //Varying Parmater "r" to
      obtain Bifurcation Diagram
22     u1 = 0;               //First Fixed Point.
23     u2 = 2*(r+1)/r;      //Second Fixed Point.
24     f1 = (r+1);         //f'(u) at u1
25     f2 = -(r+1);        //f'(u) at u2
26     set(gca()," auto_clear"," off")    //hold on
27     set(gca()," grid",[2,5])
28
29     //u(double dot) = f'(u) = (r+1) - r*u
30
31     if (f1>0) then       //Unstable Fixed Point.
32         plot2d(r+1,u1,style=-2)
33     end
34     if (f1<0) then      //Stable Fixed Point.
35         plot2d(r+1,u1,style=-3)
36     end
37     if (f2>0) then      //Unstable Fixed Point.

```

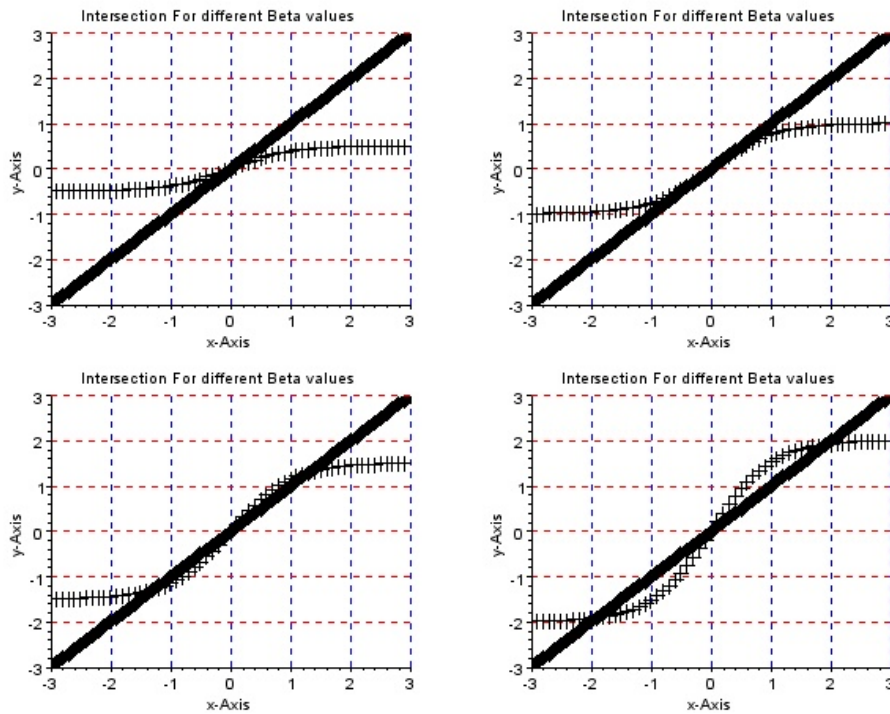


Figure 3.5: Pitchfork Bifurcation

```

38     plot2d(r+1,u2,style=-2)
39     end
40     if (f2<0) then //Stable Fixed Point.
41         plot2d(r+1,u2,style=-3)
42     end
43     xtitle(" Bifurcation Diagram", "x-Axis -(r+1)", "u
         * -fix points")
44 end
45 set(gca()," auto_clear", "on") //hold off
46 disp("Clearly from the Bifurcation Diagram we see
         that r=-1 is the bifurcation point.")
47 //end of Example

```

---

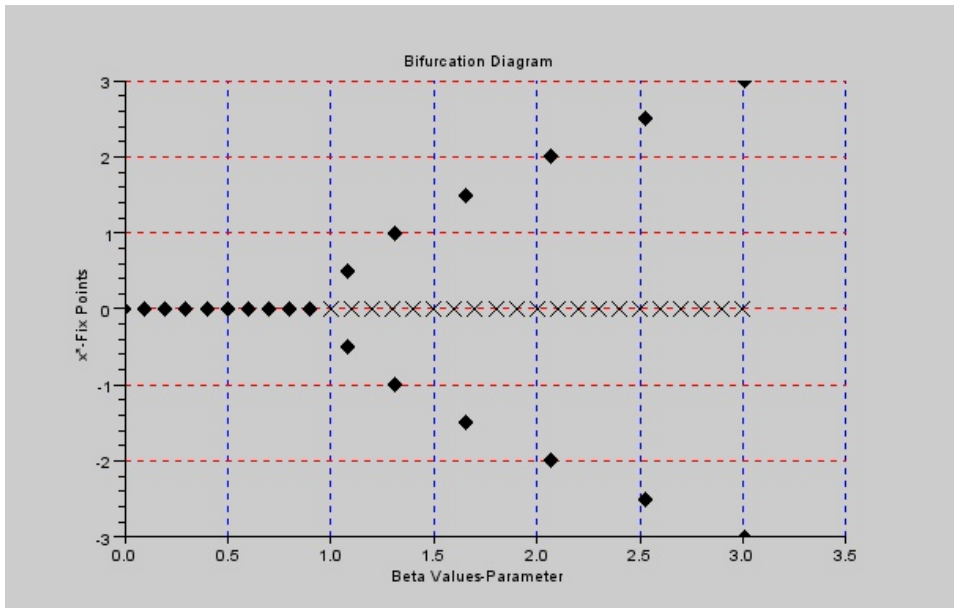


Figure 3.6: Pitchfork Bifurcation

#### Scilab code Exa 3.4.1 Pitchfork Bifurcation

```

1 //Example 3.4.1 Page 57
2 //Non-Linear Dynamics and Chaos, First Indian
  Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","on") //hold off
9
10 for (B=0.5:0.5:2) //Capital "B" is denoting Beta.

```

```

11     x=-3:0.1:3;
12     y=x;           //To plot x=y line.
13
14     figure;
15     set(gca(),"auto_clear","off")    //hold on
16     set(gca(),"grid",[2,5])
17     plot2d(x,y,style=-4)
18     plot2d(x,B*tanh(x),style=-1)
19     xtitle("Intersection For different Beta values",
            "x-Axis","y-Axis")
20
21
22 end
23 disp("From graph following points are clear:")
24 disp("1. For B<1, only origin is the fixed point.")
25 disp("2. For B>1, their are two new fixed points.")
26
27 //Stability
28 figure
29 set(gca(),"grid",[2,5])
30 for(x1 = -3:0.5:3)
31     if(x1~=0)
32         B = x1/tanh(x1);
33         plot2d(B,x1,style=-4)
34 end
35
36 for(B=0:0.1:3);
37     x1 = 0;
38 if(B<1)
39     plot2d(B,x1,style=-4)    //Stable
40 else
41     plot2d(B,x1,style=-2)    //Unstable
42 end
43 xtitle("Bifurcation Diagram","Beta Values-Parameter"
        ,"x*-Fix Points")
44 end
45 end
46 set(gca(),"auto_clear","on")

```



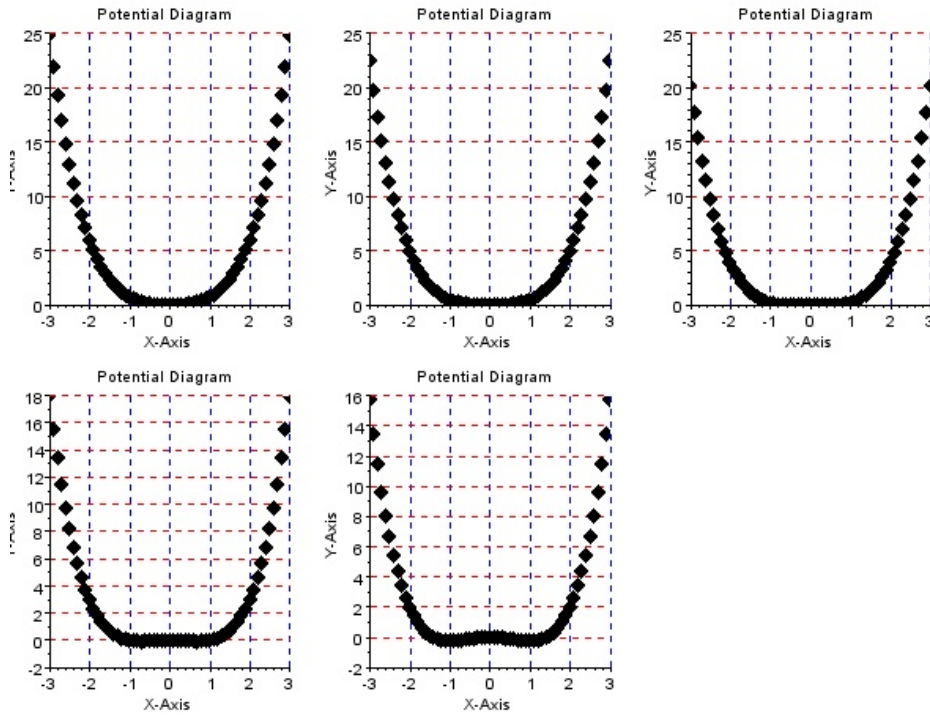


Figure 3.7: Pitchfork Bifurcation

```

47 //If B<1 then only one fixed point.
48 //end of Example.

```

---

#### Scilab code Exa 3.4.2 Pitchfork Bifurcation

```

1 //Example 3.4.2 Page 58
2 //Non-Linear Dynamics and Chaos First Indian Edition
  Print 2007
3 //Steven H. Strogatz
4 clear;
5 clc;

```

```
6 close;
7 set(gca(),"auto_clear","on") //hold off
8
9 for(r=-1:0.5:1)
10     x = -3:0.1:3;
11     V = -0.5*r*(x^2)+0.25*(x^4);
12     set(gca(),"auto_clear","off") //hold on
13     xtitle("Potential Diagram","X-Axis","Y-Axis")
14     plot2d(x,V,style=-4)
15     figure
16     set(gca(),"grid",[2,5])
17
18 end
19 set(gca(),"auto_clear","on")
20 //End of Example
```

---

# Chapter 4

## Flows on the Circle

Scilab code Exa 4.1.1 Examples and Definitions

```
1 //Example 4.1.1 Page 94
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off") //hold on
9
10 //Representing theta by O --> not zero it is
   alphabet "O"
11 //Given :- O(dot) = f(O) = sinO.
12
13 O1 = 0; //first Fix Point.
14 O2 = %pi //Second Fix Point.
15
```

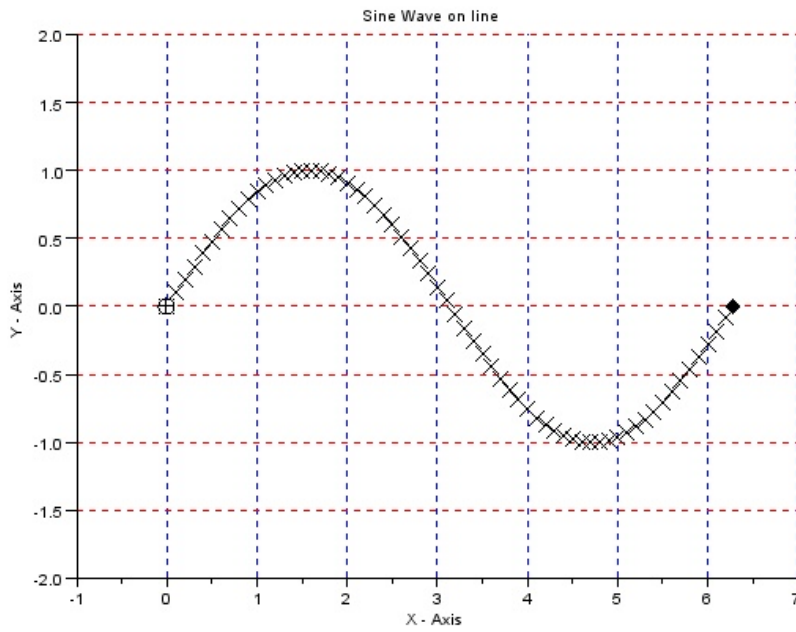


Figure 4.1: Examples and Definitions

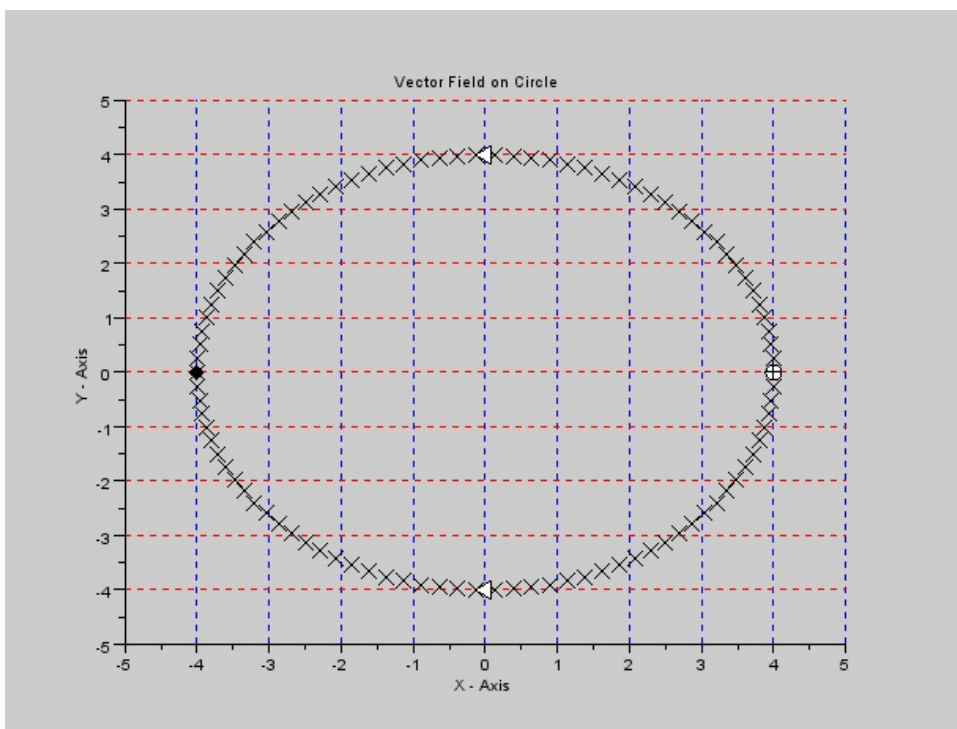


Figure 4.2: Examples and Definitions

```

16 //f(double dot) = O(double dot) = cosO.
17
18
19 ////////////////////////////////////////////////// Computations Started
    //////////////////////////////////////
20
21 f1 = cos(O1)
22 f2 = cos(O2)
23
24 if f1 > 0 then
25     disp("Zero is the Unstable Fixed Point.")
26 else
27     disp("Zero is the Stable Fixed Point.")
28 end
29
30 if f2>0 then
31     disp("Pi is the Unstable Fixed Point.")
32 else
33     disp("Pi is the Stable Fixed Point.")
34 end
35
36 //
    //////////////////////////////////////

37
38 figure(0)
39 for O = 0:0.1:2*%pi
40     f = sin(O);
41     plot2d(O,f,style=-2)
42 end
43
44 ////////////////////////////////////////////////// Figure(0) Properties
    //////////////////////////////////////
45 a=get("current_axes");//get the handle of the newly
    created axes
46 a.data_bounds=[-1,-2;7,2];
47 set(gca(),"grid",[2,5])
48 plot2d(0,0,style=-3) //Showing Unstable-

```

```

    Fixed Point with plus inside a circle
49 plot2d(2*%pi,0,style=-4)           //Showing Stable
    Fixed Point with diamond
50 xtitle("Sine Wave on line","X - Axis","Y - Axis")
51 //
    //////////////////////////////////////
52
53
54 exec circle.sci           //function to draw circle is
    executed
55 figure(1)                 //Graphic Window(1)
56 circle([0 0],4,50)        //Circle is drawn with (0,0)
    as center , radius=4.
57
58 ////////////////////////////////////// figure Properties
    //////////////////////////////////////
59 a=get("current_axes");//get the handle of the newly
    created axes
60 a.data_bounds=[-5,-5;5,5];
61 set(gca(),"grid",[2,5])
62 plot2d(0,4,style=-13)     //Showing Counter-
    Clockwise Direction
63 plot2d(0,-4,style=-13)   //Showing Counter-
    Clockwise Direction
64 plot2d(4,0,style=-3)     //Showing Unstable-
    Fixed Point with plus inside a circle
65 plot2d(-4,0,style=-4)    //Showing Stable
    Fixed Point with diamond
66 xtitle("Vector Field on Circle","X - Axis","Y - Axis
    ")
67 //
    //////////////////////////////////////
68
69 //End of Example_4-1-1.

```

---

check Appendix AP 1 for dependency:

circle.sci

### Scilab code Exa 4.1.2 Examples and Definitions

```
1 //Example 4.1.2 Page 94
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8
9 disp("A Vector Field on the Circle is a Rule :-")
10 disp("That assigns a unique velocity vector to each
    point on the circle.")
11
12 disp("For this particular example Theta=0 and Theta
    =2*pi are same points...")
13 disp("on the circle , but with different velocities."
    )
14 disp("Thus it cannot be regarded as the vector field
    on the circle.")
15
16
17 //End of Example_4.1.2
```

---

### Scilab code Exa 4.3.1 Nonuniform Oscillator



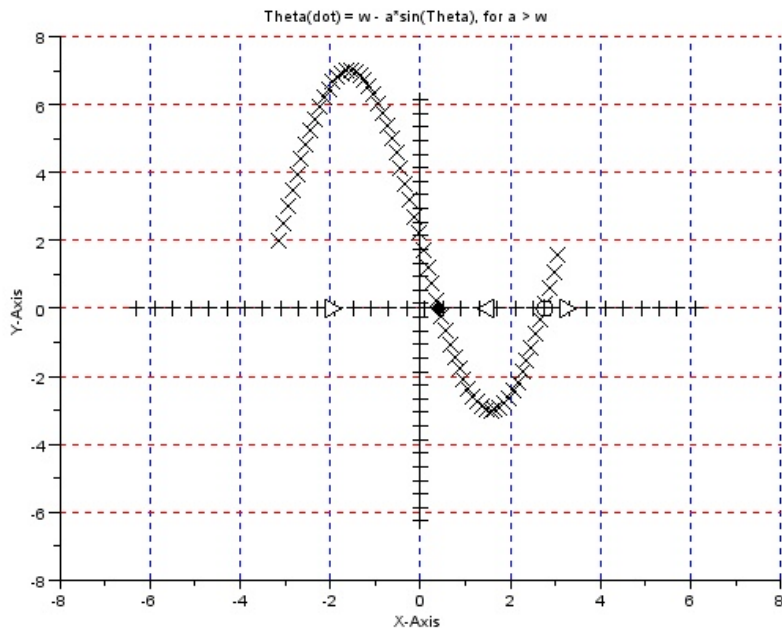


Figure 4.3: Nonuniform Oscillator

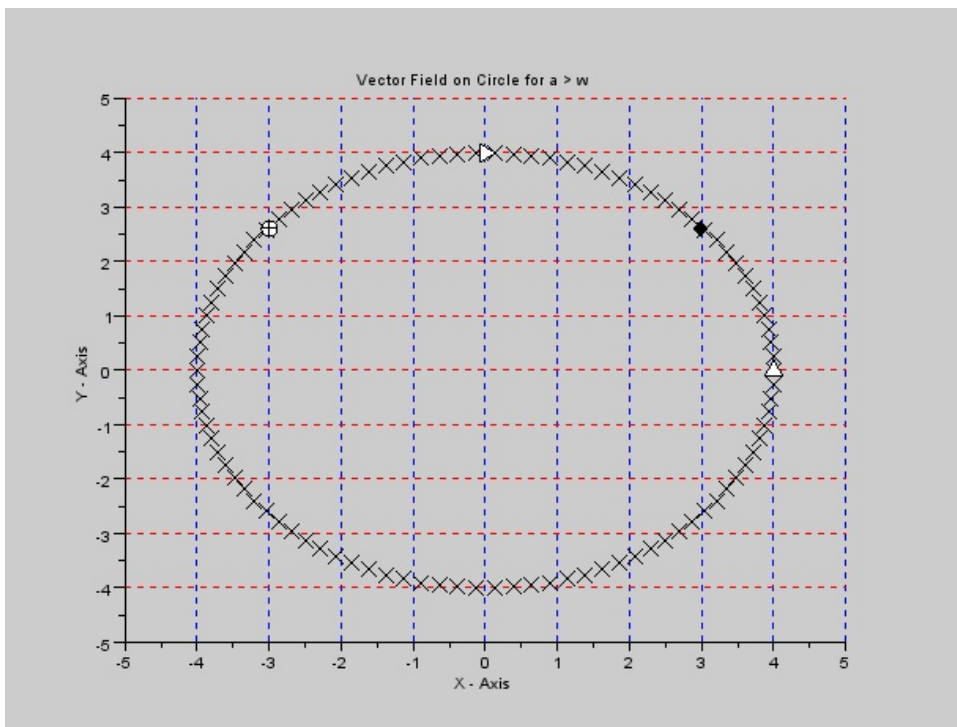


Figure 4.4: Nonuniform Oscillator

```

1 //Example 4.3.1 Page 97
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;
8 mtlb_hold on
9
10 // Theta(dot) = f = w - a*sin(Theta)
11 //a>w
12
13 //Lets take w=2, a=5;
14
15 for Theta=-%pi:0.1:%pi
16     f=2 - (5*sin(Theta));           //f = w-a*sin(Theta
   )
17     plot2d(Theta,f,style=-2)
18 end
19
20 /////////////////////////////////// Figure Characteristics
   ///////////////////////////////////
21     set(gca(),"grid",[2,5])         //Grid on
22
23 for Theta=-2*%pi:0.4:2*%pi
24     plot2d(Theta,0,style=-1)        //Just to plot X
   -Axis.
25     plot2d(0,Theta,style=-1)       //Just to plot Y
   -Axis.
26 end
27 plot2d(0.41,0,style=-4)            //Just to Show
   that the Fixed point is Stable.
28 plot2d(2.75,0,style=-3)           //Just to Show
   that the Fixed point is UnStable.
29 plot2d(1.5,0,style=-13)           //Just to Show
   the Flow.
30 plot2d(-2,0,style=-12)            //Just to Show the

```

```

Flow.
31 plot2d(3.2,0,style=-12)           //Just to Show
    the Flow.
32     xtitle("Theta(dot) = w - a*sin(Theta), for a >
        ; w", "X-Axis", "Y-Axis")
33 //
    //////////////////////////////////////
34
35 exec circle.sci           //function to draw circle is
    executed
36 figure(1)                //Graphic Window(1)
37 circle([0 0],4,50)       //Circle is drawn with (0,0)
    as center , radius=4.
38
39 ////////////////////////////////////// figure Properties
    //////////////////////////////////////
40 a=get("current_axes"); //get the handle of the newly
    created axes
41 a.data_bounds=[-5,-5;5,5];
42 set(gca(),"grid",[2,5])
43 plot2d(0,4,style=-12)     //Showing Vector
    Fields on Circle
44 plot2d(4,0,style=-6)     //Showing Vector
    Fields on Circle
45 plot2d(-3,2.6,style=-3)  //Showing
    Unstable-Fixed Point with plus inside a circle
46 plot2d(3,2.6,style=-4)  //Showing Stable
    Fixed Point with diamond
47 xtitle("Vector Field on Circle for a > w", "X -
    Axis", "Y - Axis")
48 //
    //////////////////////////////////////
49
50 //End of Example_4-3-1.

```

---

check Appendix [AP 1](#) for dependency:

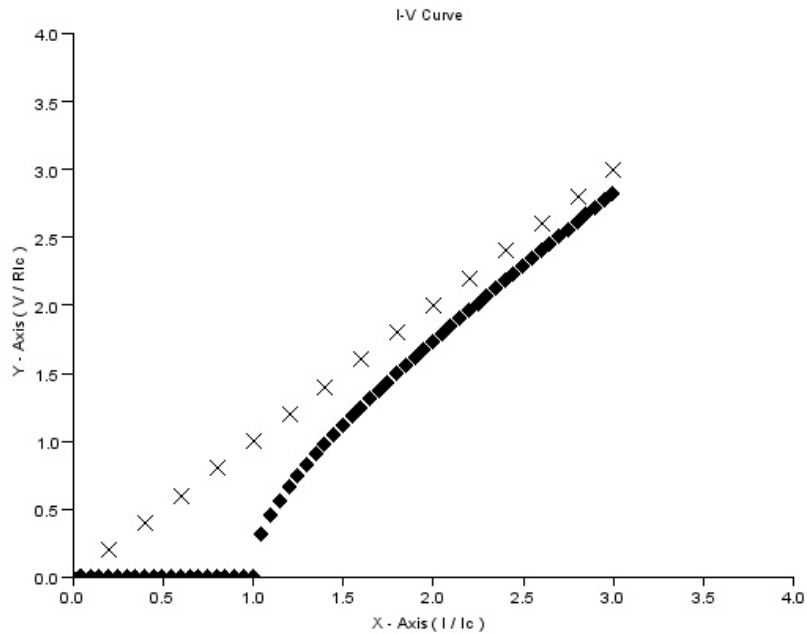


Figure 4.5: Superconducting Josephson Junctions

circle.sci

**Scilab code Exa 4.6.1 Superconducting Josephson Junctions**

```

1 //Example 4.6.1 Page 110
2 //Non-Linear Dynamics and Chaos, First Indian
  Edition Print 2007
3 //Steven H. Strogatz
4
5 clear;
6 clc;
7 close;

```

```

8 set(gca(),"auto_clear","off")           //hold on
9
10 //After mathematical calculations they found:
11
12 //<V> = 0 for I<=Ic
13 //<V> = Ic*R*sqrt((I/Ic)^2 - 1) for I>Ic
14
15 //Let V/RIc = Y
16 //Let I/Ic = X
17
18 //so, we have
19 //Y = 0 for X<=1.
20 //Y = sqrt((X^2)-1) for X>1.
21
22 for X=0:0.05:3
23     if X<=1 then
24         Y=0;
25         plot2d(X,Y,style=-4)
26     else
27         Y=sqrt((X^2)-1);
28         plot2d(X,Y,style=-4)
29     end
30 end
31
32 for X=0:0.2:3
33     Y=X;
34     plot2d(X,Y,style=-2)
35 end
36
37 a=get("current_axes");//get the handle of the newly
    created axes
38 a.data_bounds=[0,0;4,4];
39 xtitle("I-V Curve","X - Axis ( I / Ic )","Y - Axis (
    V / RIc )")
40
41 //End of Example 4.6.1

```

---

# Chapter 5

## Linear Systems

Scilab code Exa 5.1.1 Definitions and Examples

```
1 clear;
2 clc;
3 close;
4 mtlb_hold on
5
6 circle([0 0],4,50) //Circle is drawn with (0,0)
   as center , radius=4.
7 circle([0,0],2,50)
8
9 a=get("current_axes"); //get the handle of
   the newly created axes
10 a.data_bounds=[-5,-5;5,5];
11
12 for x = -4:2:4
13     for v = -4:2:4
14         if(x==0) & (v==0)
15             plot2d(x,v,style=-4) //if x=0 and
```

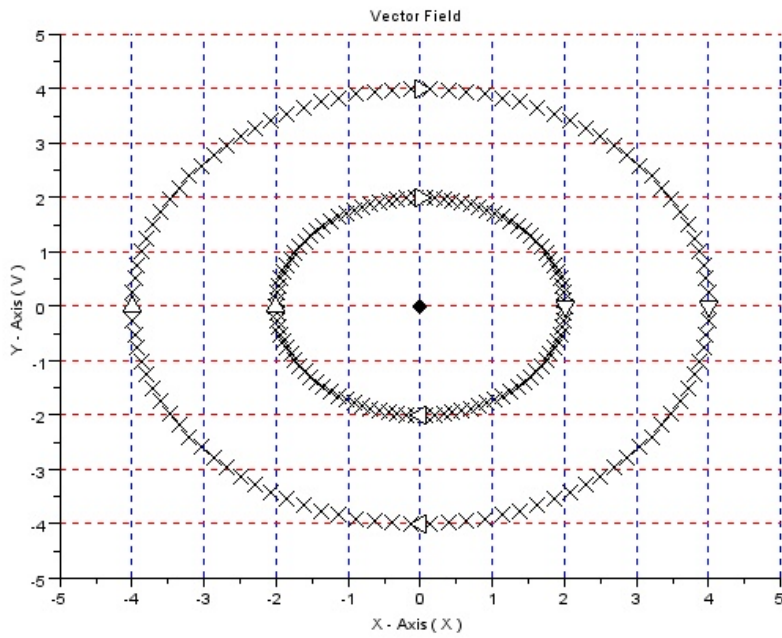


Figure 5.1: Definitions and Examples



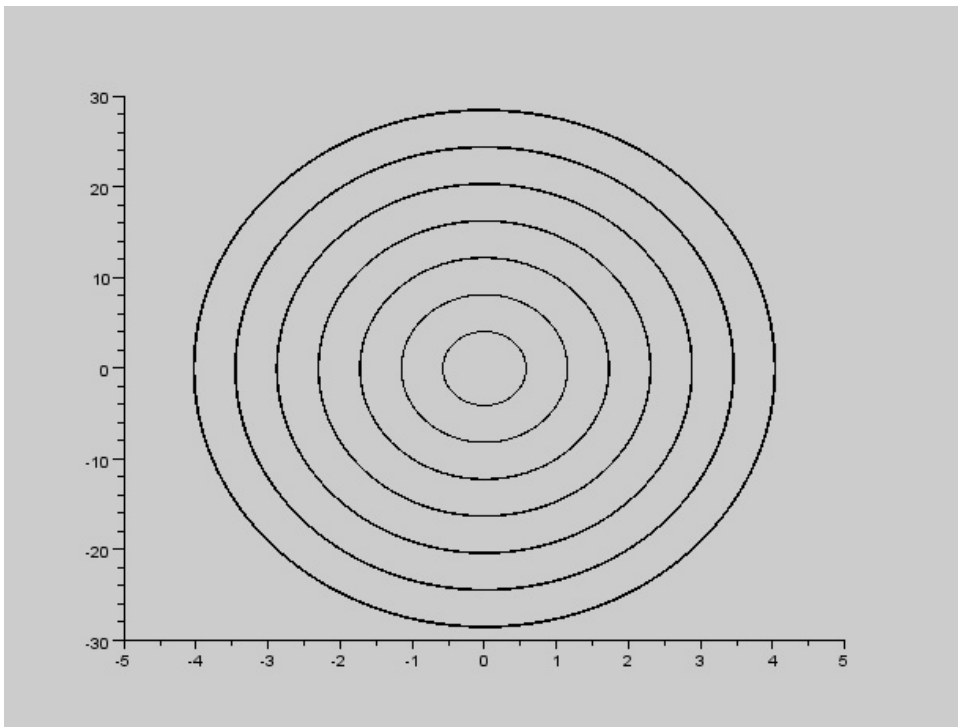


Figure 5.2: Definitions and Examples

```

                                v=0 then x(dot)and v(d
                                ot) are also zero , thus Fixed point.
16     end
17     if(x==0) & amp; (v>0)
18         plot2d(x,v,style=-12)
19     end
20     if(x==0) & amp; (v<0)
21         plot2d(x,v,style=-13)
22     end
23     if(v==0) & amp; (x>0)
24         plot2d(x,v,style=-7)
25     end
26     if(v==0) & amp; (x<0)
27         plot2d(x,v,style=-6)
28     end
29 end
30 end
31
32 a=get("current_axes");           //get the handle
    of the newly created axes
33 a.data_bounds=[-5,-5;5,5];
34 xtitle("Vector Field","X - Axis ( X )","Y - Axis ( V
    )")
35     set(gca(),"grid",[2,5])           //Grid on
36
37 figure
38 function xd=linear511(t,x)
39     xd(1)=x(2);                       //x(dot);    x(2)
        means v.
40     xd(2)=-50*x(1);                   //v(dot);    x(1)
        means x.; Taking w^2=50;
41 endfunction
42 bound=[-4,-4,4,4];                   //Bounds of x-axis and y
    -axis as [xmin ymin xma
        x ymax], change them
        according to your needs.
43 nrect=15;                             //increase it to get

```

```

        more number of curves , i.e
                                                . more information
        will be available.
44  set(gca(),"auto_clear","off")           //hold on
45  x=linspace(bound(1),bound(3),nrect);
46  y=linspace(bound(2),bound(4),nrect);
47  x0=[];
48
49  for i=1:15
50      x0=[x(i);y(i)];
51      t0=0;
52      t=0:0.01:3000;
53      xout=ode(x0,t0,t,linear511);
54      plot2d(xout(1,:),xout(2,:));
55  end

```

---

check Appendix [AP 1](#) for dependency:

circle.sci

### Scilab code Exa 5.1.2 Definitions and Examples

```

1  clear;
2  clc;
3  close;
4  set(gca(),"auto_clear","off")           //hold on
5
6  a=-1;                                     //Change a to get different figures
    ; Not work for a=0 as z1 is not defined as slope
    is infinty
7  i=1;j=1;
8  for x=-10:1:10
9      if (x<>0) then                         //x<>0 because z1
        not defined at x=0;

```

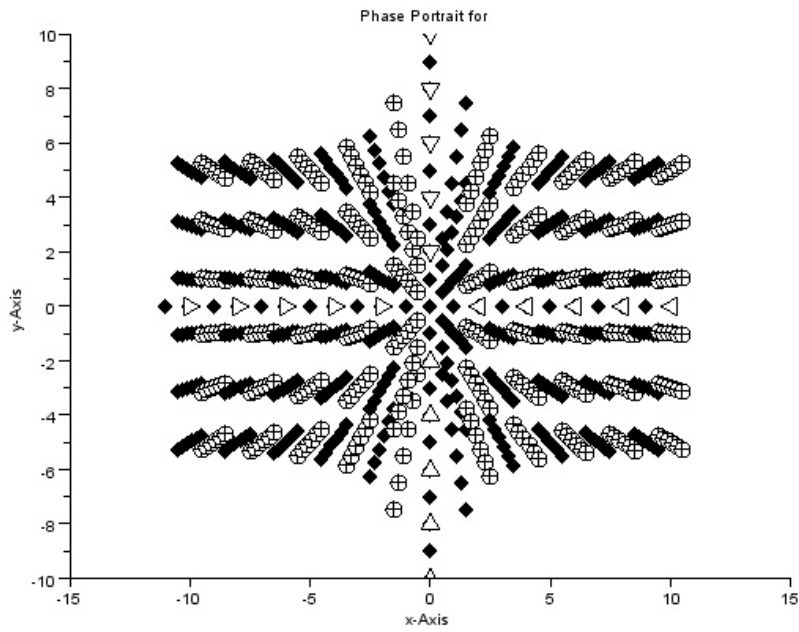


Figure 5.3: Definitions and Examples

```

10     j=1;
11     for y=-5:2:5
12         x1 = a*x;                //x1=x(dot)
13         y1 = -y;                //y1=y(dot)
14         mat1(i,j)=x;
15         mat2(i,j)=y;
16         //plot2d(x1,y1,style=-4)
17         z1(i,j) = y1./x1;        //tangent matrix;
                                   calculating slope at every point.
18         j=j+1;
19     end
20     i=i+1;
21 end
22 end
23
24
25
26 ////////////////////////////////////////////////////////////////// Phase Portrait computations
   Started //////////////////////////////////////////////////////////////////
27
28 //Algorithm: (y-y0)=m*(x-x0) equation is used.
29 //A straight line is drawn in every small interval
   surrounding x.
30 i=1;
31     for yo=-5:2:5
32         for x=-10.5:0.2:-9.5
33             y=(z1(1,i)*(x+10))+yo;
34             plot2d(x,y,style=-4)
35         end
36         i=i+1;
37     end
38
39     // for x=-11:0.1:-9
40         // y=(0.25*(x+10))-3;
41         // plot2d(x,y,style=-3)
42     //end
43         // for x=-11:0.1:-9
44         //y=(0.25*(x+10))-1;

```

```

45         //plot2d(x,y,style=-1)
46     //end
47
48     i=1;
49     for yo=-5:2:5
50         for x=-9.5:0.2:-8.5
51             y=(z1(2,i)*(x+9))+yo;
52             plot2d(x,y,style=-3)
53         end
54         i=i+1;
55     end
56
57     i=1;
58     for yo=-5:2:5
59         for x=-8.5:0.2:-7.5
60             y=(z1(3,i)*(x+8))+yo;
61             plot2d(x,y,style=-4)
62         end
63         i=i+1;
64     end
65
66     i=1;
67     for yo=-5:2:5
68         for x=-7.5:0.2:-6.5
69             y=(z1(4,i)*(x+7))+yo;
70             plot2d(x,y,style=-3)
71         end
72         i=i+1;
73     end
74
75     i=1;
76     for yo=-5:2:5
77         for x=-6.5:0.2:-5.5
78             y=(z1(5,i)*(x+6))+yo;
79             plot2d(x,y,style=-4)
80         end
81         i=i+1;
82     end

```

```

83
84 i=1;
85     for yo=-5:2:5
86         for x=-5.5:0.2:-4.5
87             y=(z1(6,i)*(x+5))+yo;
88             plot2d(x,y,style=-3)
89         end
90         i=i+1;
91     end
92
93 i=1;
94     for yo=-5:2:5
95         for x=-4.5:0.2:-3.5
96             y=(z1(7,i)*(x+4))+yo;
97             plot2d(x,y,style=-4)
98         end
99         i=i+1;
100    end
101
102
103 i=1;
104     for yo=-5:2:5
105         for x=-3.5:0.2:-2.5
106             y=(z1(8,i)*(x+3))+yo;
107             plot2d(x,y,style=-3)
108         end
109         i=i+1;
110    end
111
112
113 i=1;
114     for yo=-5:2:5
115         for x=-2.5:0.2:-1.5
116             y=(z1(9,i)*(x+2))+yo;
117             plot2d(x,y,style=-4)
118         end
119         i=i+1;
120    end

```

```

121
122 i=1;
123     for yo=-5:2:5
124         for x=-1.5:0.2:-0.5
125             y=(z1(10,i)*(x+1))+yo;
126             plot2d(x,y,style=-3)
127         end
128         i=i+1;
129     end
130
131
132 i=1;
133     for yo=-5:2:5
134         for x=0.5:0.2:1.5
135             y=(z1(11,i)*(x-1))+yo;
136             plot2d(x,y,style=-4)
137         end
138         i=i+1;
139     end
140
141 i=1;
142     for yo=-5:2:5
143         for x=1.5:0.2:2.5
144             y=(z1(12,i)*(x-2))+yo;
145             plot2d(x,y,style=-3)
146         end
147         i=i+1;
148     end
149
150
151 i=1;
152     for yo=-5:2:5
153         for x=2.5:0.2:3.5
154             y=(z1(13,i)*(x-3))+yo;
155             plot2d(x,y,style=-4)
156         end
157         i=i+1;
158     end

```



```

159
160
161 i=1;
162     for yo=-5:2:5
163         for x=3.5:0.2:4.5
164             y=(z1(14,i)*(x-4))+yo;
165             plot2d(x,y,style=-3)
166         end
167         i=i+1;
168     end
169
170 i=1;
171     for yo=-5:2:5
172         for x=4.5:0.2:5.5
173             y=(z1(15,i)*(x-5))+yo;
174             plot2d(x,y,style=-4)
175         end
176         i=i+1;
177     end
178 i=1;
179     for yo=-5:2:5
180         for x=5.5:0.2:6.5
181             y=(z1(16,i)*(x-6))+yo;
182             plot2d(x,y,style=-3)
183         end
184         i=i+1;
185     end
186
187
188 i=1;
189     for yo=-5:2:5
190         for x=6.5:0.2:7.5
191             y=(z1(17,i)*(x-7))+yo;
192             plot2d(x,y,style=-4)
193         end
194         i=i+1;
195     end
196

```

```

197 i=1;
198     for yo=-5:2:5
199         for x=7.5:0.2:8.5
200             y=(z1(18,i)*(x-8))+yo;
201             plot2d(x,y,style=-3)
202         end
203     i=i+1;
204 end
205
206 i=1;
207     for yo=-5:2:5
208         for x=8.5:0.2:9.5
209             y=(z1(19,i)*(x-9))+yo;
210             plot2d(x,y,style=-4)
211         end
212     i=i+1;
213 end
214
215
216 i=1;
217     for yo=-5:2:5
218         for x=9.5:0.2:10.5
219             y=(z1(20,i)*(x-10))+yo;
220             plot2d(x,y,style=-3)
221         end
222     i=i+1;
223 end
224
225 //////////////// Phase Portrait Computations Completed
226 ////////////////
227
228 //////////////// Drawing X-Axis
229 for x=-11:9
230     y=0;
231     plot2d(x,y,style=-4)
232 end
233

```

```

234 //////////////// Drawing Y-Axis
235 for y=-10:10
236     x=0;
237     plot2d(x,y,style=-4)
238
239 end
240
241 // flow lines on x-axis
242 y=0;
243
244 for x=-10:2:10
245     x1=a*x;
246     if(x1>0)
247         plot2d(x,y,style=-12);
248     elseif(x1<0)
249         plot2d(x,y,style=-13);
250     else
251         plot2d(x,y,style=-4);
252     end
253 end
254
255
256 // Flow lines on y-axis
257 x=0;
258
259 for y=-10:2:10
260     y1=-y;
261     if(y1>0)
262         plot2d(x,y,style=-6);
263     elseif(y1<0)
264         plot2d(x,y,style=-7);
265     else
266         plot2d(x,y,style=-4);
267     end
268 end
269
270
271     xtitle("Phase Portrait for","x-Axis","y-Axis")

```

---

Scilab code Exa 5.2.1 Classification of Linear Systems

```
1 clear;
2 clc;
3 close;
4
5 //Get the corresponding matrix equation.
6
7 A = [1 1;4 -2]
8
9 [R,E]=spec(A)
10
11 //where E corresponds to eigen values
12 //And R corresponds to eigen vectors.
13
14 //Note that 0.9701425 = -4*0.2425356
15 //And we can assume 0.7071068 to be 1.
```

---

Scilab code Exa 5.2.2 Classification of Linear Systems

```
1 clear;
2 clc;
3 close;
4 set(gca(),"auto_clear","off") //hold on
5
6 for x=-20:0.5:20
7     y1 = x;
8     y=0;
9     x1=0;
```

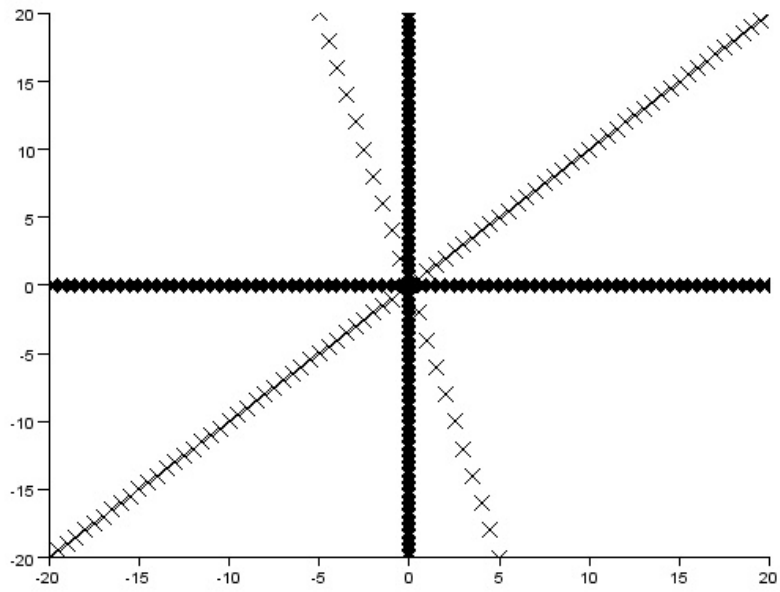


Figure 5.4: Classification of Linear Systems

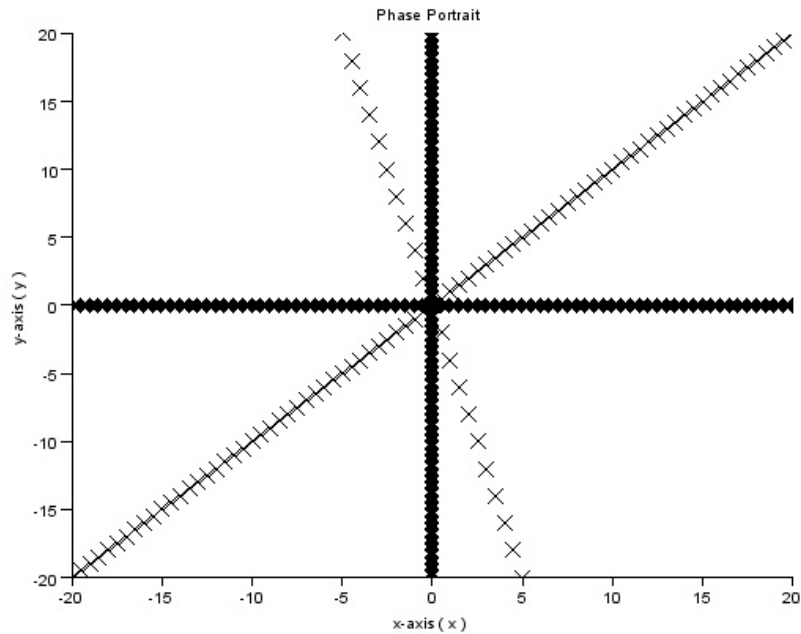


Figure 5.5: Classification of Linear Systems

```

10 //h1=acosh(x);
11 plot2d(x,y1,style=-2)
12 plot2d(x,y,style=-4)
13 plot2d(x1,x,style=-4)
14
15 end
16
17 for x=-5:0.5:5
18     y2=-4*x;
19     plot2d(x,y2,style=-2)
20 end

```

---

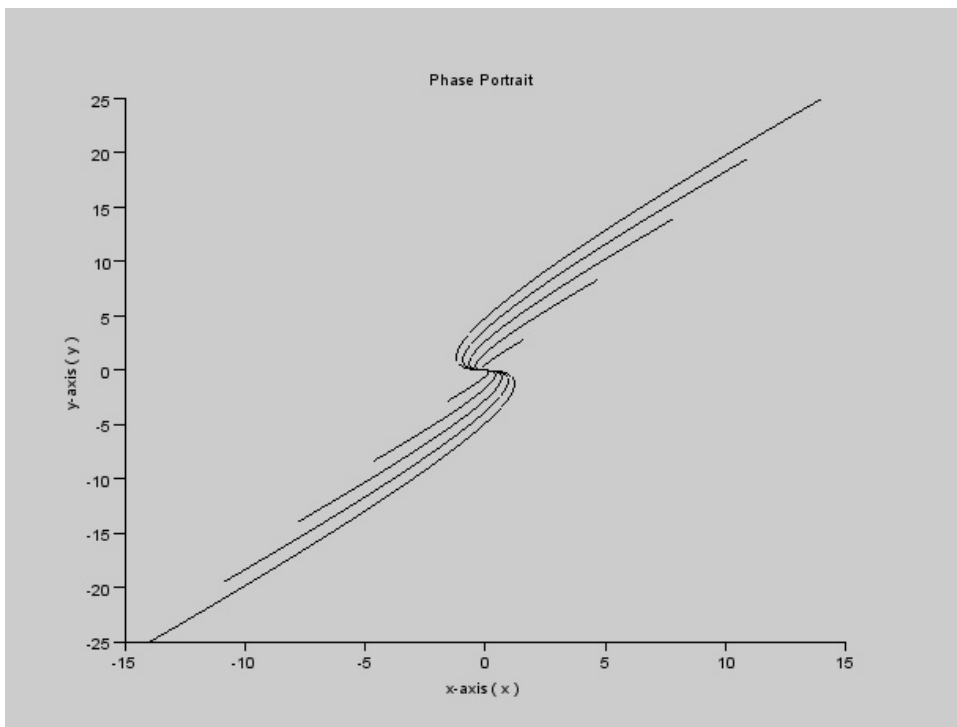


Figure 5.6: Classification of Linear Systems

### Scilab code Exa 5.2.3 Classification of Linear Systems

```
1 clear;
2 clc;
3 close;
4 set(gca()," auto_clear"," off")           //hold on
5
6 //Let lambda1<lambda2<0
7
8 for x=-20:0.5:20
9     y1 = x;
10    y=0;
11    x1=0;
12    plot2d(x,y1,style=-2)
13    plot2d(x,y,style=-4)
14    plot2d(x1,x,style=-4)
15 end
16
17 for x=-5:0.5:5
18    y2=-4*x;
19    plot2d(x,y2,style=-2)
20 end
21 xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')
22 figure
23 function xd=linear523(t,x)
24    xd(1)=-x(1)-x(2);           //x(dot);    x(2) means
        y.
25    xd(2)=0*x(1)-2*x(2);       //y(dot);    x(1) means
        x.;
26 endfunction
27 bound=[-14,-25,14,25];       //Bounds of x-axis
        and y-axis as [xmin ymin xmax ymax], change
        them according to your needs.
```



```

28  nrect=10;          //increase it to get more number of
    curves , i.e. more information will be available
    .
29  set(gca()," auto_clear"," off")          //hold on
30  x=linspace(bound(1),bound(3),nrect);
31  y=linspace(bound(2),bound(4),nrect);
32  x0=[];
33
34  for i=1:10
35      x0=[x(i);y(i)];
36      t0=0;
37      t=0:0.01:3000;
38      xout=ode(x0,t0,t,linear523);
39      plot2d(xout(1,:),xout(2,:));
40  end
41  xtitle('Phase Portrait ','x-axis ( x )','y-axis ( y
    )')

```

---

#### Scilab code Exa 5.2.4 Classification of Linear Systems

```

1  clear;
2  clc;
3  close;
4
5  disp("When the eigen values are complex , then the
    fixed point is either :")
6  disp(" 1.      Center , or")
7  disp(" 2.      Spiral.")
8
9  function xd=linear524(t,x)
10     xd(1)=0*x(1)+x(2);          //x(dot);    x(2)

```

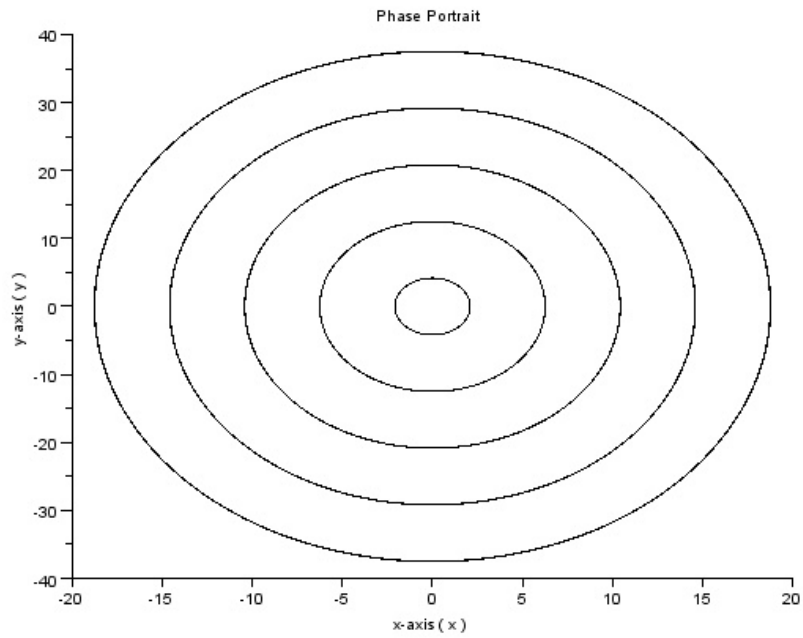


Figure 5.7: Classification of Linear Systems

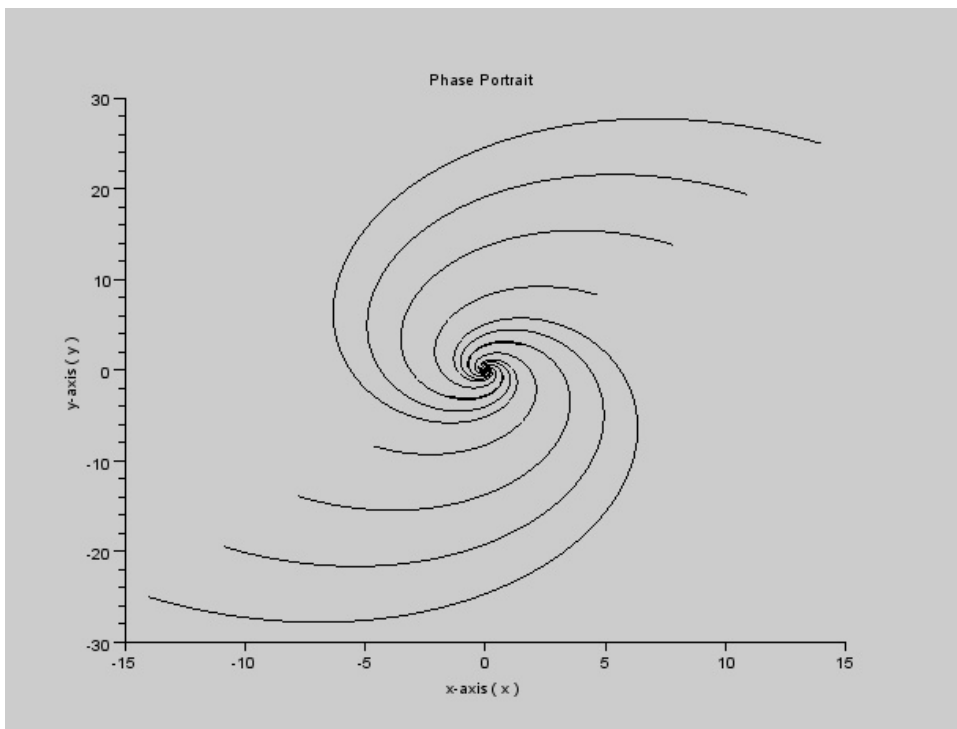


Figure 5.8: Classification of Linear Systems

```

        means y.
11     xd(2)=-4*x(1)+0*x(2);      //y(dot);   x(1) means
        x.;
12     endfunction
13     bound=[-14,-25,14,25];      //Bounds of x-axis
        and y-axis as [xmin ymin xmax ymax], change
        them according to your needs.
14     nrect=10;      //increase it to get more number of
        curves, i.e. more information will be available
        .
15     set(gca(),"auto_clear","off")      //hold on
16     x=linspace(bound(1),bound(3),nrect);
17     y=linspace(bound(2),bound(4),nrect);
18     x0=[];
19
20     for i=1:10
21         x0=[x(i);y(i)];
22         t0=0;
23         t=0:0.01:3000;
24         xout=ode(x0,t0,t,linear524);
25         plot2d(xout(1,:),xout(2,:));
26     end
27     xtitle('Phase Portrait','x-axis ( x )','y-axis (
        y )')
28     figure
29     function xd=linear5242(t,x)
30         xd(1)=-x(1)-x(2);      //x(dot);   x(2) means
        y.
31         xd(2)=4*x(1)-x(2);      //y(dot);   x(1) means x
        .;
32     endfunction
33     bound=[-14,-25,14,25];      //Bounds of x-axis
        and y-axis as [xmin ymin xmax ymax], change
        them according to your needs.
34     nrect=10;      //increase it to get more number of
        curves, i.e. more information will be available
        .
35     set(gca(),"auto_clear","off")      //hold on

```

```

36  x=linspace(bound(1),bound(3),nrect);
37  y=linspace(bound(2),bound(4),nrect);
38  x0=[];
39
40  for i=1:10
41      x0=[x(i);y(i)];
42      t0=0;
43      t=0:0.01:3000;
44      xout=ode(x0,t0,t,linear5242);
45      plot2d(xout(1,:),xout(2,:));
46  end
47  xtitle('Phase Portrait','x-axis ( x )','y-axis (
      y )')

```

---

#### Scilab code Exa 5.2.6 Classification of Linear Systems

```

1  clear;
2  clc;
3  close;
4
5  A=[1 2;3 4]
6
7  t=det(A)
8
9  disp("Since t = -2 hence the fixed point is a Stable
      fixed point.")

```

---

#### Scilab code Exa 5.2.7 Classification of Linear Systems

```

1  clear;
2  clc;
3  close;
4

```

```
5 A=[2 1;3 4]
6
7 t=det(A)
8 tau=trace(A)
9
10 disp("Since t>0 and (tau)^2 - 4*t =16 >0 hence the
      fixed point is a node.")
11 disp("It is unstable since tau>0.")
```

---

# Chapter 6

## Phase Plane

Scilab code Exa 6.1.1 Phase Portraits

```
1 clear;
2 clc;
3 close;
4 set(gca()," auto_clear", " off")           //hold on
5
6 for y=-3:0.5:5
7     x = -exp(-y);
8     x1=0;
9     plot2d(x,y,style=-2)
10    plot2d(x1,y,style=-4)
11 end
12
13
14 for x=-30:0.5:5
15     y=0;
16     plot2d(x,y,style=-4)
17 end
18
19 ////////////////////////////////////////////////// Flow //////////////////////////////////////
```

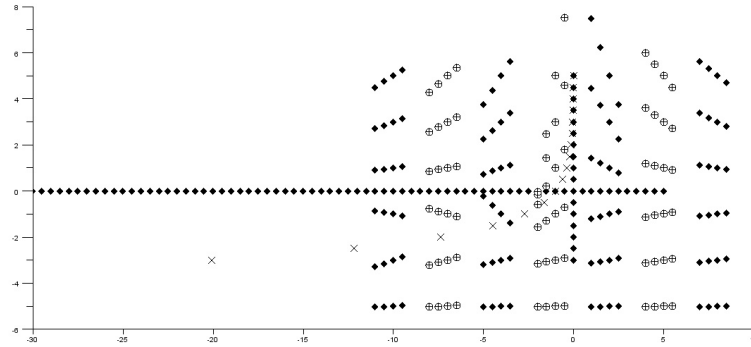


Figure 6.1: Phase Portraits

```

20
21 i=1;j=1;
22 for x=-10:3:10
23     j=1;
24     for y=-5:2:5
25         x1 = x+exp(-y);
26         y1 = -y;
27         mat1(i,j)=x;
28         mat2(i,j)=y;
29         //plot2d(x1,y1,style=-4)
30         z1(i,j) = y1./x1;
31         j=j+1;
32     end
33     i=i+1;
34 end
35
36
37 i=1;
38 for yo=-5:2:5
39     for x=-11:0.5:-9.5
40         y=(z1(1,i)*(x+10))+yo;
41         plot2d(x,y,style=-4)
42     end

```



```

43         i=i+1;
44     end
45
46         // for x=-11:0.1:-9
47         //     y=(0.25*(x+10))-3;
48         //     plot2d(x,y,style=-3)
49         //end
50         //         for x=-11:0.1:-9
51         //     y=(0.25*(x+10))-1;
52         //     plot2d(x,y,style=-1)
53         //end
54
55     i=1;
56     for yo=-5:2:5
57         for x=-8:0.5:-6.5
58             y=(z1(2,i)*(x+7))+yo;
59             plot2d(x,y,style=-3)
60         end
61         i=i+1;
62     end
63
64     i=1;
65     for yo=-5:2:5
66         for x=-5:0.5:-3.5
67             y=(z1(3,i)*(x+4))+yo;
68             plot2d(x,y,style=-4)
69         end
70         i=i+1;
71     end
72
73     i=1;
74     for yo=-5:2:5
75         for x=-2:0.5:-0.5
76             y=(z1(4,i)*(x+1))+yo;
77             plot2d(x,y,style=-3)
78         end
79         i=i+1;
80     end

```

```

81
82 i=1;
83     for yo=-5:2:5
84         for x=1:0.5:2.5
85             y=(z1(5,i)*(x-2))+yo;
86             plot2d(x,y,style=-4)
87         end
88         i=i+1;
89     end
90
91 i=1;
92     for yo=-5:2:5
93         for x=4:0.5:5.5
94             y=(z1(6,i)*(x-5))+yo;
95             plot2d(x,y,style=-3)
96         end
97         i=i+1;
98     end
99
100 i=1;
101     for yo=-5:2:5
102         for x=7:0.5:8.5
103             y=(z1(7,i)*(x-8))+yo;
104             plot2d(x,y,style=-4)
105         end
106         i=i+1;
107     end

```

---

### Scilab code Exa 6.3.1 Fixed Points and Linearization

```

1 clear;
2 clc;
3 close;

```

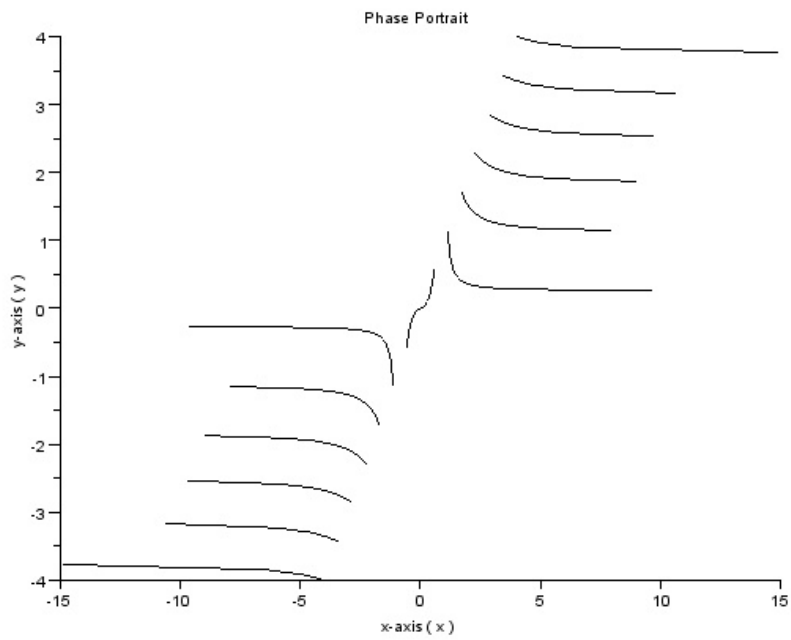


Figure 6.2: Fixed Points and Linearization

```

4
5 A1=[-1 0;0 -2] //Jacobian at (0,0)
6
7 t1=det(A1)
8
9 tau1=trace(A1)
10 d1=((tau1)^2) - 4*t1
11
12 A2=[2 0;0 -2] //Jacobian at (1,0) and (-1,0)
13 t2=det(A2)
14
15 disp("So from Chapter 5, we come to following
      conclusion :")
16 disp("1. As t1>0 and tau1<.0 and d1>0 --> Thus (0,0)
      is Stable Node.")
17 disp("2. As t2<0 --> Thus, only possibility is
      Saddle points.")
18
19 function xd=linear611(t,x)
20     xd(1)=-x(1)+x(1)^3; //x(dot); x(2)
      means y.
21     xd(2)=-2*x(2); //y(dot); x(1) means x.;
22 endfunction
23 bound=[-4,-4,4,4]; //Bounds of x-axis and y
      -axis as [xmin ymin xmax ymax], change them
      according to your needs.
24 nrect=15; //increase it to get more number of
      curves, i.e. more information will be available
      .
25 set(gca(),"auto_clear","off") //hold on
26 x=linspace(bound(1),bound(3),nrect);
27 y=linspace(bound(2),bound(4),nrect);
28 x0=[];
29
30 for i=1:15
31     x0=[x(i);y(i)];
32     t0=0;
33     t=0:0.01:3000;

```

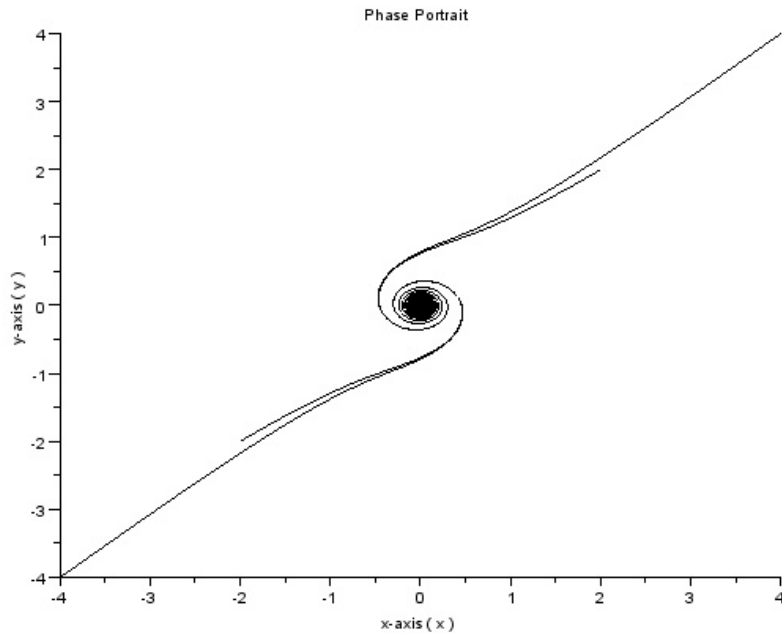


Figure 6.3: Fixed Points and Linearization

```

34     xout=ode(x0,t0,t,linear611);
35     plot2d(xout(1,:),xout(2,:));
36     end
37     xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
        )')

```

---

### Scilab code Exa 6.3.2 Fixed Points and Linearization

```

1 clear;

```

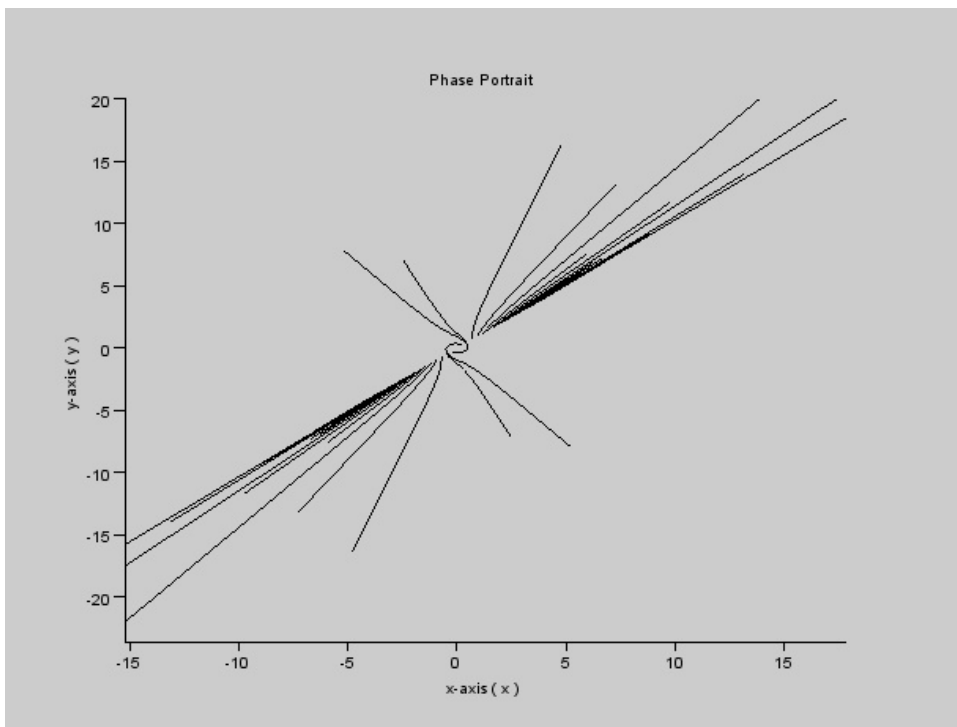


Figure 6.4: Fixed Points and Linearization

```

2  clc;
3  close;
4
5  //On linearization we got the following Jacobian "A"
6
7  A=[0 -1;1 0];
8  t=det(A);
9  tau=trace(A);
10
11 disp("Since tau = 0 and t>0, thus from chapter 5,
      (0,0) is a Center")
12 disp("And this does not depends on value of a ")
13
14 a=-1;
15 function xd=linear632(t,x)
16     xd(1)=-x(2)+ a*x(1)*(x(1)^2+x(2)^2);           //x
      (dot);    x(2) means y.
17     xd(2)=x(1)+a*x(2)*(x(1)^2+x(2)^2);           //y(dot);
      x(1) means x.;
18 endfunction
19 bound=[-4,-4,4,4];           //Bounds of x-axis and y
      -axis as [xmin ymin xmax ymax], change them
      according to your needs.
20 nrect=5;           //increase it to get more number of
      curves, i.e. more information will be available
      .
21 set(gca(),"auto_clear","off")           //hold on
22 x=linspace(bound(1),bound(3),nrect);
23 y=linspace(bound(2),bound(4),nrect);
24 x0=[];
25
26 for i=1:5
27     x0=[x(i);y(i)];
28     t0=0;
29     t=0:0.01:3000;
30     xout=ode(x0,t0,t,linear632);
31     plot2d(xout(1,:),xout(2,:));
32 end

```

```

33     xtitle('Phase Portrait', 'x-axis ( x )', 'y-axis (
        y )')
34     figure
35     a=1;
36     function xd=linear6322(t,x)
37         xd(1)=-x(2)+ a*x(1)*(x(1)^2+x(2)^2);           //x
            (dot);    x(2) means y.
38         xd(2)=x(1)+a*x(2)*(x(1)^2+x(2)^2);           //y(dot);
            x(1) means x.;
39     endfunction
40     bound=[-4,-4,4,4];           //Bounds of x-axis and y
        -axis as [xmin ymin xmax ymax], change them
        according to your needs.
41     nrect=35;           //increase it to get more number of
        curves, i.e. more information will be available
        .
42     set(gca()," auto_clear"," off")           //hold on
43     x=linspace(bound(1),bound(3),nrect);
44     y=linspace(bound(2),bound(4),nrect);
45     x0=[];
46
47     for i=1:35
48         x0=[x(i);y(i)];
49         t0=0;
50         t=0:0.01:3000;
51         xout=ode(x0,t0,t,linear6322);
52         plot2d(xout(1,:),xout(2,:));
53     end
54     xtitle('Phase Portrait', 'x-axis ( x )', 'y-axis (
        y )')

```

---



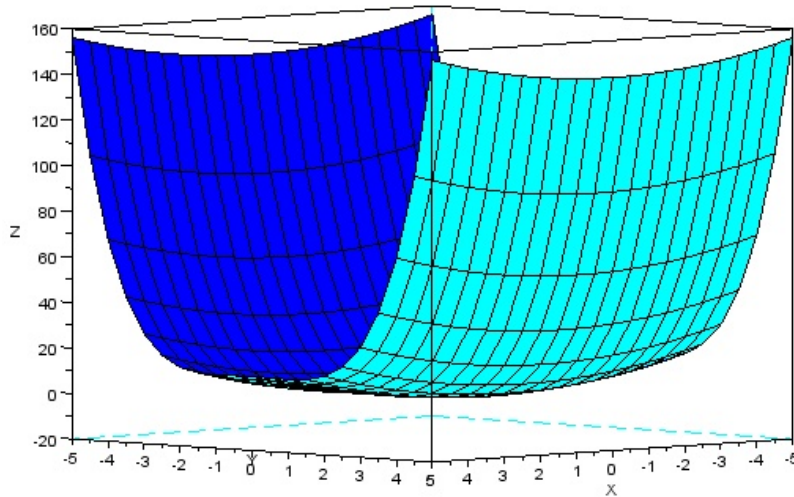


Figure 6.5: Conservative Systems

### Scilab code Exa 6.5.2 Conservative Systems

```

1  clear;
2  clear;
3  clc;
4  close;
5  set(gca()," auto_clear"," off")           //hold on
6
7  //Get x(dot) = y;    y(dot)=x-(x^3);
8
9  A1 = [0 1;1 0]           //Jacobian at (0,0)
10 t1=det(A1)
11
12 A2 = [0 1;-2 0]         //Jacobian at (1,0) and
    (-1,0).
13 t2=det(A2)
14 tau2=trace(A2)
15
16 disp(" Since , t1=-1, thus (0,0) is a Saddle Point.");

```

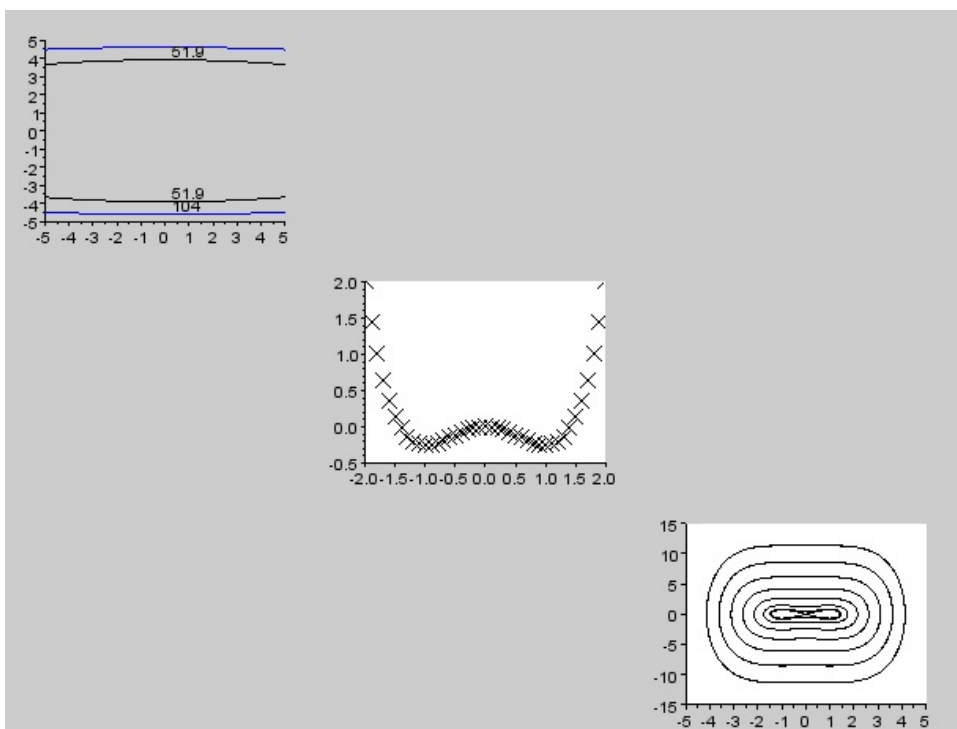


Figure 6.6: Conservative Systems

```

17 disp("As t2=2 and tau2=0, thus (1,0) and (-1,0) are
    Centers.");
18
19
20 [x y]=meshgrid(-5:0.5:5,-5:0.5:5);
21 Nx=21;
22 Ny=21;
23
24
25 for i=1:21
26     for j=1:21
27         E(i,j) = ((1/2)*(y(i,j)^2)) -((1/2)*(x(i,j)
                ^2)) + ((1/4)*(x(i,j)^4));
28         //contour(x,y,E,3);
29     end
30 end
31
32 l=-5:0.5:5;
33 k=-5:0.5:5;
34 plot3d(l,k,E);
35 figure
36 subplot(331)
37 contour2d(l,k,E,2);
38
39 subplot(335)
40 for x=-2:0.1:2
41     V=((-1/2)*(x^2))+((1/4)*(x^4));
42     plot2d(x,V,style=-2)
43 end
44 subplot(339)
45 function xd=linear652(t,x)
46     xd(1)=x(2);
47     xd(2)=x(1)-x(1)^3;
48     //x(dot);    x(2) means y.
49     //y(dot);    x(1) means x.;
50 endfunction
51 bound=[-4,-4,4,4];           //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them

```

```

    according to your needs.
52  nrect=16;      //increase it to get more number of
    curves, i.e. more information will be available
    .
53  set(gca()," auto_clear"," off")      //hold on
54  x=linspace(bound(1),bound(3),nrect);
55  y=linspace(bound(2),bound(4),nrect);
56  x0=[];
57
58  for i=1:35
59      x0=[x(i);y(i)];
60      t0=0;
61      t=0:0.01:3000;
62      xout=ode(x0,t0,t,linear652);
63      plot2d(xout(1,:),xout(2,:));
64  end
65  xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')

```

---

### Scilab code Exa 6.6.1 Reversible Systems

```

1  clear;
2  clear;
3  clc;
4  close;
5  //set(gca()," auto_clear"," off")      //hold on
6
7  //Obtain the Jacobian A:
8  //A = [df/dx df/dy; dg/dx dg/dy];      where f=x(
    dot),g=y(dot)
9  //df/dx=partial derivative of "f" w.r.t. "x"
10 //Thus, A=[0 1-3(y^2); -1 -2*y];
11
12 A1=[0 1;-1 0]      //Jacobian at origin
13 t1=det(A1)

```

```

14 tau1=trace(A1)
15
16 disp("Since t1>0 and tau1=0 thus origin is a linear
      center.")
17 disp("Furthermore, the system is reversible, Thus by
      theorem 6.6.1-->")
18 disp("We conclude that the origin is a nonlinear
      center.")
19
20 A2=[0 -2;-1 -2]           //Jacobian at (-1,1)
21 t2=det(A2)
22
23 A3=[0 -2;-1 2]           //Jacobian at (-1,-1)
24 t3=det(A3)
25
26 disp("Since t2=t3=-2, Thus (-1,1) and (-1,-1) are
      Saddle Points.")

```

---

### Scilab code Exa 6.6.3 Reversible Systems

```

1 clear;
2 clear;
3 clc;
4 close;
5 //obtain the General Jacobian "A";
6
7 x1= %pi/2; y1= %pi/2;
8 x2= %pi/2; y2= -%pi/2;
9 x3= -%pi/2; y3= %pi/2;
10 x4= -%pi/2; y4= -%pi/2;
11
12 A1 = [2*sin(x1) sin(y1); sin(x1) 2*sin(y1)]
13 t1=det(A1)
14 tau1=trace(A1)
15 d1 = ((tau1)^2) - 4*t1

```

```

16 A2 = [2*sin(x2) sin(y2); sin(x2) 2*sin(y2)]
17 t2=det(A2)
18 tau2=trace(A2)
19 d2 = ((tau2)^2) - 4*t2
20 A3 = [2*sin(x3) sin(y3); sin(x3) 2*sin(y3)]
21 t3=det(A3)
22 tau3=trace(A3)
23 d3 = ((tau3)^2) - 4*t3
24 A4 = [2*sin(x4) sin(y4); sin(x4) 2*sin(y4)]
25 t4=det(A4)
26 tau4=trace(A4)
27 d4 = ((tau4)^2) - 4*t4
28
29
30 disp("From the above information we come to
      following conclusion:")
31 disp(" 1.      (pi/2,pi/2) --> Unstable Node.")
32 disp(" 2.      (pi/2,-pi/2) --> Saddle.")
33 disp(" 3.      (-pi/2,pi/2) --> Saddle.")
34 disp(" 4.      (-pi/2,-pi/2) --> Stable Node.")

```

---

# Chapter 7

## Limit Cycles

Scilab code Exa 7.1.1 A Simple Limit Cycle

```
1 clear;
2 clear;
3 clc;
4 close;
5 set(gca(),"auto_clear","off") //hold on
6
7 // Note r=-1 is not the fixed point as radius can't
  be negative.
8
9 r1=0; //First Fixed Point
10 r2=1; //Second Fixed Point
11
12 for r=0:0.05:1.1
13     f=r*(1-(r^2));
14     plot2d(r,f,style=-2)
15 end
16
```

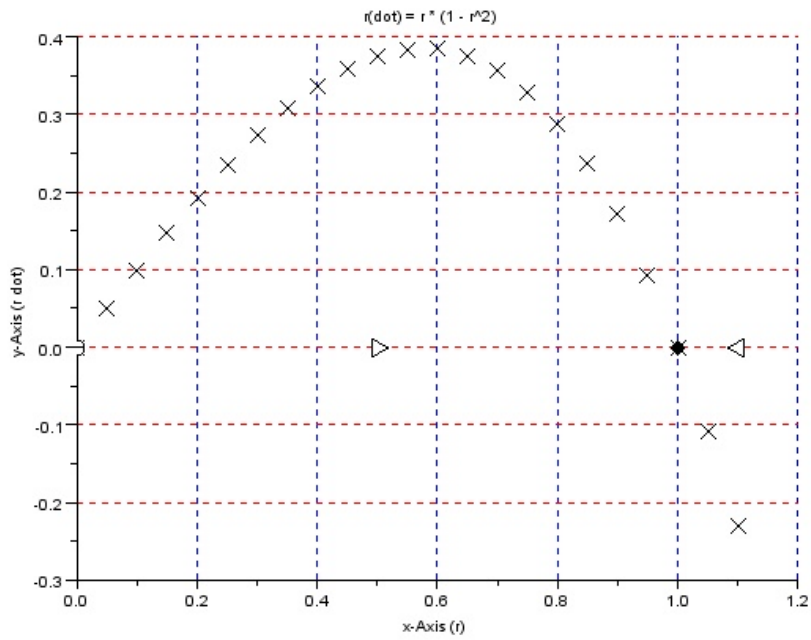


Figure 7.1: A Simple Limit Cycle



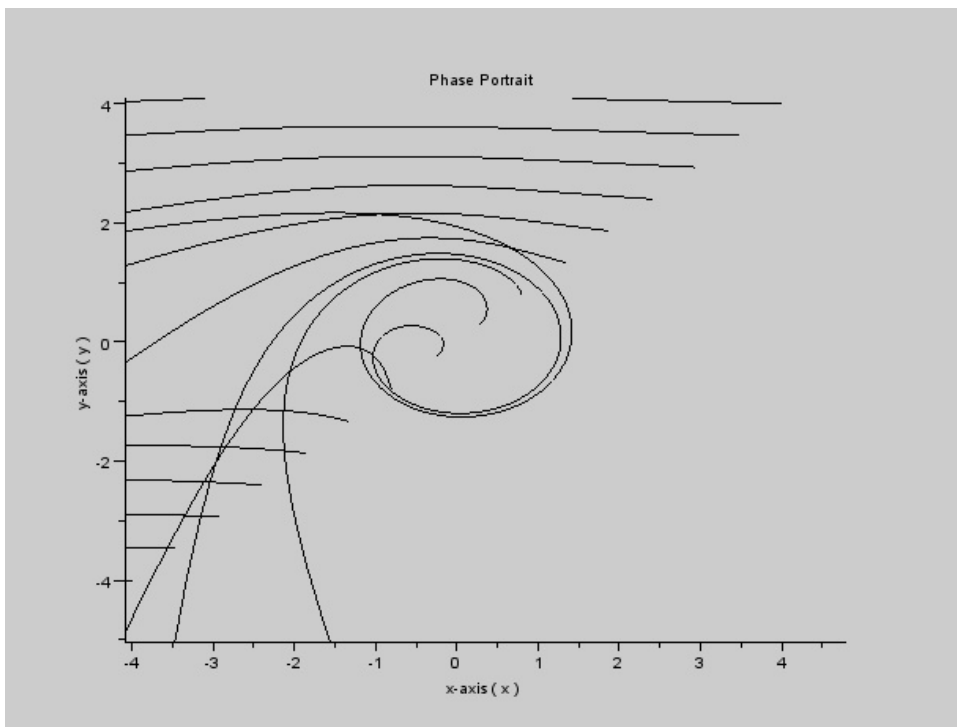


Figure 7.2: A Simple Limit Cycle

```

17 set(gca(),"grid",[2,5])           //Grid on
18 plot2d(0,0,style=-3)
19 plot2d(1,0,style=-4)
20 plot2d(0.5,0,style=-12)
21 plot2d(1.1,0,style=-13)
22 xtitle("r(dot) = r * (1 - r^2)", "x-Axis (r)", "y-Axis
      (r dot)")
23 figure
24 function xd=linear711(t,x)
25     xd(1)=(cos(t)*sqrt(x(1)^2+x(2)^2)*(1-x(1)^2-x(2)
      ^2))-(sqrt(x(1)^2+x(2)^2)*sin(t));
26     xd(2)=(sin(t)*sqrt(x(1)^2+x(2)^2)*(1-x(1)^2-x(2)
      ^2))+(sqrt(x(1)^2+x(2)^2)*cos(t));
27         //x(dot);    x(2) means y.
28         //y(dot);    x(1) means x.;
29 endfunction
30 bound=[-4,-4,4,4];           //Bounds of x-axis and y
      -axis as [xmin ymin xmax ymax], change them
      according to your needs.
31 nrect=16;           //increase it to get more number of
      curves, i.e. more information will be available
      .
32 set(gca(),"auto_clear","off")           //hold on
33 x=linspace(bound(1),bound(3),nrect);
34 y=linspace(bound(2),bound(4),nrect);
35 x0=[];
36
37 for i=1:16
38     x0=[x(i);y(i)];
39     t0=0;
40     t=0:0.01:6000;
41     xout=ode(x0,t0,t,linear711);
42     plot2d(xout(1,:),xout(2,:));
43 end
44 xtitle('Phase Portrait','x-axis ( x )','y-axis (
      y )')

```

---

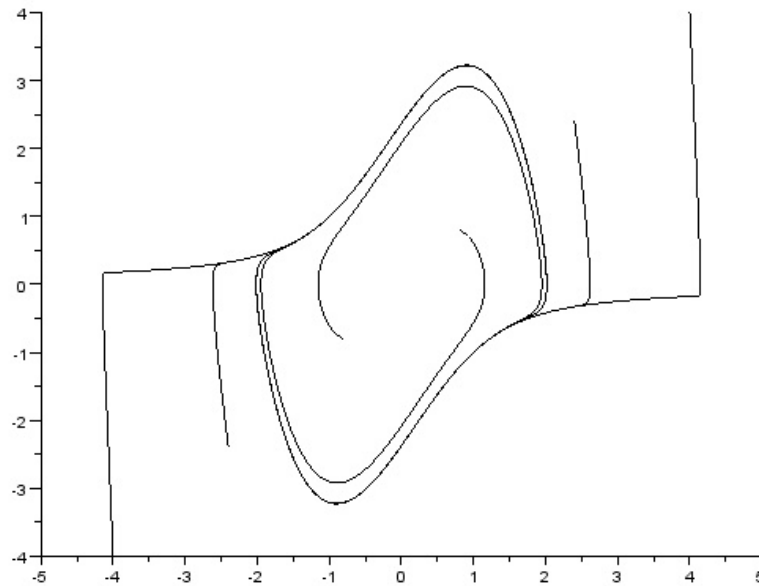


Figure 7.3: Van der Pol Oscillator

#### Scilab code Exa 7.1.2 Van der Pol Oscillator

```

1 function xd=linear712(t,x)
2     xd(1)=x(2);
3     xd(2)=1.5*x(2)*(1-x(1)^2)-x(1);
4         //x(dot);    x(2) means y.
5         //y(dot);    x(1) means x.;
6 endfunction
7 bound=[-4,-4,4,4]; //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them

```

```

    according to your needs.
8   nrect=6;      //increase it to get more number of
    curves , i.e. more information will be available
    .
9   set(gca()," auto_clear"," off")      //hold on
10  x=linspace(bound(1),bound(3),nrect);
11  y=linspace(bound(2),bound(4),nrect);
12  x0=[];
13
14  for i=1:6
15      x0=[x(i);y(i)];
16      t0=0;
17      t=0:0.01:6000;
18      xout=ode(x0,t0,t,linear712);
19      plot2d(xout(1,:),xout(2,:));
20  end
21  xtitle('Phase Portrait','x-axis ( x )','y-axis (
    y )')

```

---

### Scilab code Exa 7.2.3 Ruling Out Closed Orbits

```

1  function xd=linear723(t,x)
2      xd(1)=-x(1)+4*(x(2));      //x(dot);    x(2)
    means y.
3      xd(2)=-x(1)-(x(2)^3);      //y(dot);    x(1) means
    x.;
4  endfunction
5  bound=[-8,-8,8,8];      //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them
    according to your needs.
6  nrect=10;      //increase it to get more number of
    curves , i.e. more information will be available
    .

```

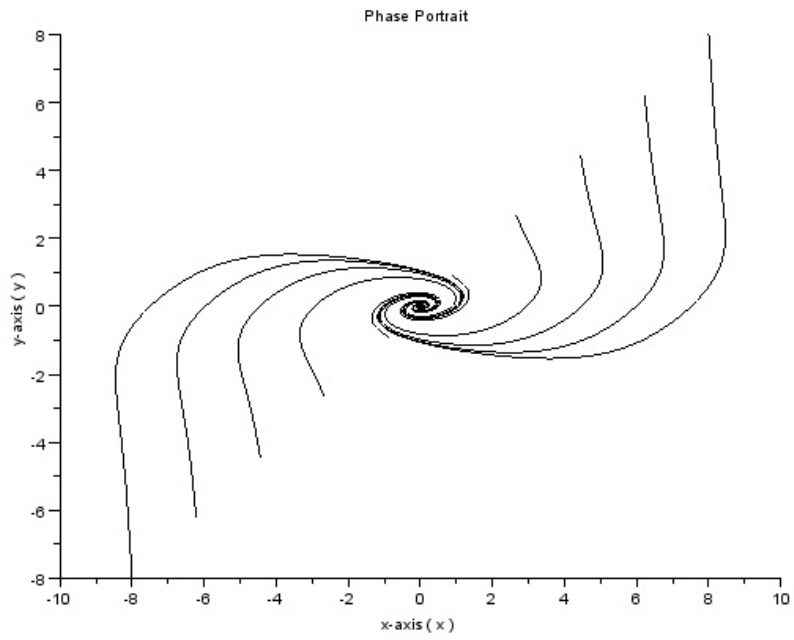


Figure 7.4: Ruling Out Closed Orbits

```

7   set(gca()," auto_clear"," off")           //hold on
8   x=linspace(bound(1),bound(3),nrect);
9   y=linspace(bound(2),bound(4),nrect);
10  x0=[];
11
12  for i=1:10
13      x0=[x(i);y(i)];
14      t0=0;
15      t=0:0.01:6000;
16      xout=ode(x0,t0,t,linear723);
17      plot2d(xout(1,:),xout(2,:));
18  end
19  xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')
20  disp("Change x and y bounds and also observe by
    changing nrect-->")
21  disp("We observe no close orbits for this system."
    )

```

---

### Scilab code Exa 7.2.5 Ruling Out Closed Orbits

```

1  function xd=linear725(t,x)
2      xd(1)=(x(2));           //x(dot);    x(2) means y.
3      xd(2)=-x(1)-x(2)+x(1)^2+x(2)^2;   //y(dot);
        x(1) means x.;
4  endfunction
5  bound=[-15,-15,15,15];     //Bounds of x-axis
        and y-axis as [xmin ymin xmax ymax], change
        them according to your needs.
6  nrect=22;                 //increase it to get more number of
        curves, i.e. more information will be available
        .
7  set(gca()," auto_clear"," off")           //hold on

```

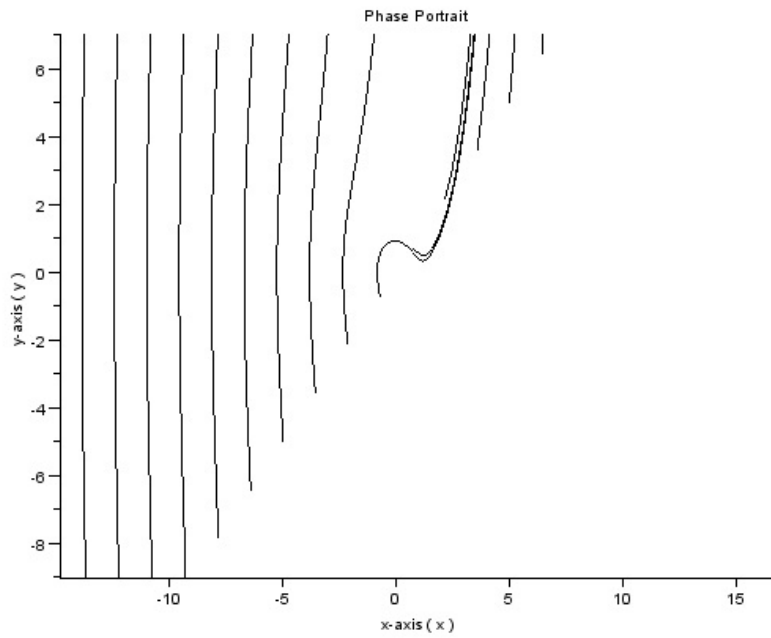


Figure 7.5: Ruling Out Closed Orbits

```

8   x=linspace(bound(1),bound(3),nrect);
9   y=linspace(bound(2),bound(4),nrect);
10  x0=[];
11
12  for i=1:22
13      x0=[x(i);y(i)];
14      t0=0;
15      t=0:0.01:6000;
16      xout=ode(x0,t0,t,linear725);
17      plot2d(xout(1,:),xout(2,:));
18  end
19  xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')

```

---

### Scilab code Exa 7.3.1 Poincare Bendixson Theorem

```

1  function xd=linear731(t,x)
2   xd(1)=(cos(t)*sqrt(x(1)^2+x(2)^2)*(1-x(1)^2-x(2)^2)
      )+(cos(t)*1*sqrt(x(1)^2+x(2)^2)*cos(t))-(sqrt(x
      (1)^2+x(2)^2)*sin(t));
3   xd(2)=(sin(t)*sqrt(x(1)^2+x(2)^2)*(1-x(1)^2-x(2)
      ^2)+(sin(t)*1*sqrt(x(1)^2+x(2)^2)*cos(t))-(
      sqrt(x(1)^2+x(2)^2)*cos(t));
4       //x(dot);   x(2) means y.
5       //y(dot);   x(1) means x.;
6  endfunction
7  bound=[-4,-4,4,4];           //Bounds of x-axis and y
      -axis as [xmin ymin xmax ymax], change them
      according to your needs.
8  nrect=12;                   //increase it to get more number of
      curves, i.e. more information will be available
9  set(gca(),"auto_clear","off")           //hold on

```



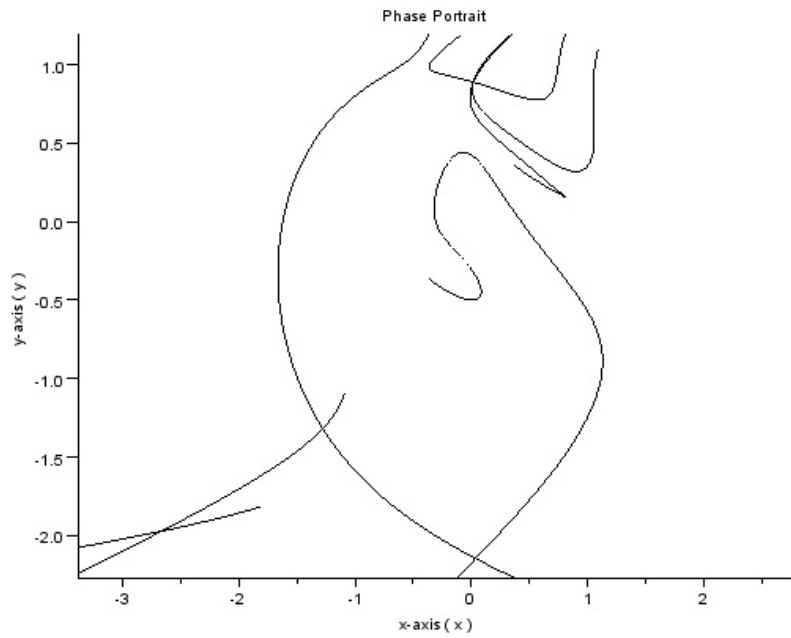


Figure 7.6: Poincare Bendixson Theorem

```

10  x=linspace(bound(1),bound(3),nrect);
11  y=linspace(bound(2),bound(4),nrect);
12  x0=[];
13
14  for i=1:12
15      x0=[x(i);y(i)];
16      t0=0;
17      t=0:0.01:3000;
18      xout=ode(x0,t0,t,linear731);
19      plot2d(xout(1,:),xout(2,:));
20  end
21  xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')

```

---

### Scilab code Exa 7.3.2 Poincare Bendixson Theorem

```

1  function xd=linear732(t,x)
2      xd(1)=-x(1)+2*x(2)+(x(1)^2)*x(2);
3      xd(2)=(4)-2*x(2)-(x(1)^2)*x(2);
4  endfunction
5  bound=[-4,-4,4,4];          //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them
    according to your needs.
6  nrect=22;                  //increase it to get more number of
    curves, i.e. more information will be available
    .
7  set(gca(),"auto_clear","off") //hold on
8  x=linspace(bound(1),bound(3),nrect);
9  y=linspace(bound(2),bound(4),nrect);
10 x0=[];
11
12 for i=1:22
13     x0=[x(i);y(i)];

```

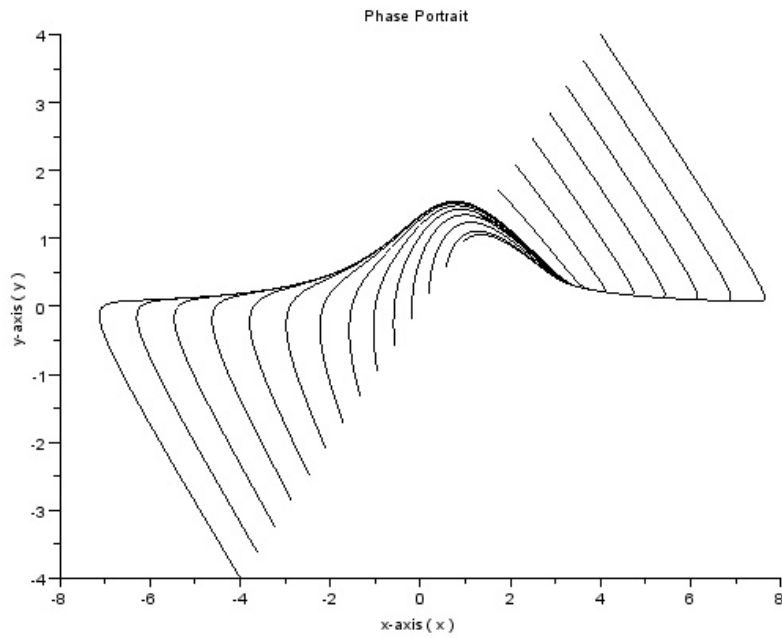


Figure 7.7: Poincare Bendixson Theorem

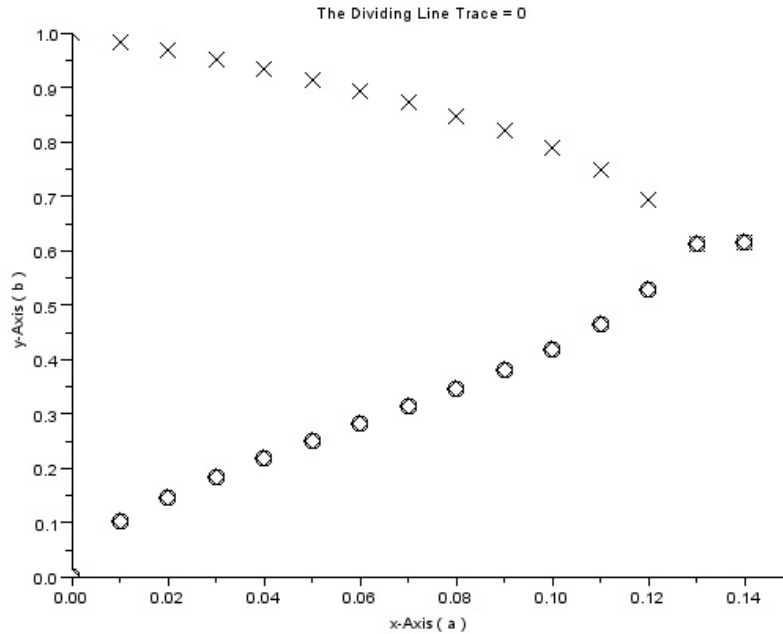


Figure 7.8: Poincare Bendixson Theorem

```

14     t0=0;
15     t=0:0.01:3000;
16     xout=ode(x0,t0,t,linear732);
17     plot2d(xout(1,:),xout(2,:));
18 end
19     xtitle('Phase Portrait','x-axis ( x )','y-axis (
    y )')
20 disp("From the figure we can clearly observe that
    the trajectories are giving it a limit cycle
    shape.")

```

---

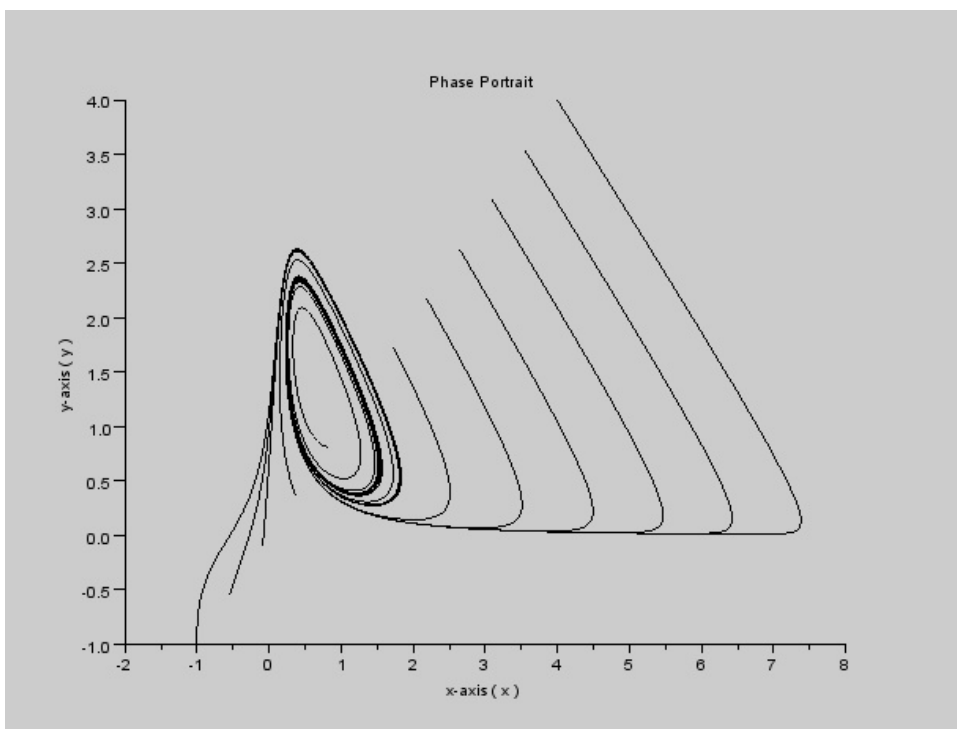


Figure 7.9: Poincare Bendixson Theorem

### Scilab code Exa 7.3.3 Poincare Bendixson Theorem

```
1 clear;
2 clear;
3 clc;
4 close;
5 set(gca(),"auto_clear","off")           //hold on
6
7 //Fixed Point is Stable for determinant(D) > 0 and
  trace(T) < 0.
8 //To get limit cycle oscillations there should not
  be any fixed point
9 //near to the limit cycle.
10
11 //Thus, only possibility left as D > 0, to have T >
  0.
12
13 //Curve between a and b :
14
15 for a=0:0.01:0.14
16     b1=+sqrt(0.5*(1-2*a+sqrt(1-8*a)));
17     b2=+sqrt(0.5*(1-2*a-sqrt(1-8*a)));
18     b3=-sqrt(0.5*(1-2*a+sqrt(1-8*a)));
19     b4=-sqrt(0.5*(1-2*a-sqrt(1-8*a)));
20     plot2d(a,b1,style=-2)
21     plot2d(a,b2,style=-3)
22     plot2d(a,b2,style=-4)
23     plot2d(a,b2,style=-5)
24
25
26 end
27 xtitle("The Dividing Line Trace = 0", "x-Axis ( a )",
  "y-Axis ( b )")
28
```

```

29 figure
30 a=0.08;
31 b=0.6;
32 function xd=linear733(t,x)
33     xd(1)=-x(1)+a*x(2)+(x(1)^2)*x(2);
34     xd(2)=b-a*x(2)-((x(1)^2)*x(2));
35 endfunction
36 bound=[-1,-1,4,4];           //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them
    according to your needs.
37 nrect=12;                   //increase it to get more number of
    curves, i.e. more information will be available
    .
38 set(gca(),"auto_clear","off") //hold on
39 x=linspace(bound(1),bound(3),nrect);
40 y=linspace(bound(2),bound(4),nrect);
41 x0=[];
42
43 for i=1:12
44     x0=[x(i);y(i)];
45     t0=0;
46     t=0:0.01:3000;
47     xout=ode(x0,t0,t,linear733);
48     plot2d(xout(1,:),xout(2,:));
49 end
50 xtitle('Phase Portrait','x-axis ( x )','y-axis (
    y )')

```

---

#### Scilab code Exa 7.4.1 Lienard Systems

```

1 function xd=linear741(t,x)
2     xd(1)=x(2);
3     xd(2)=1.5*x(2)*(1-x(1)^2)-x(1);

```

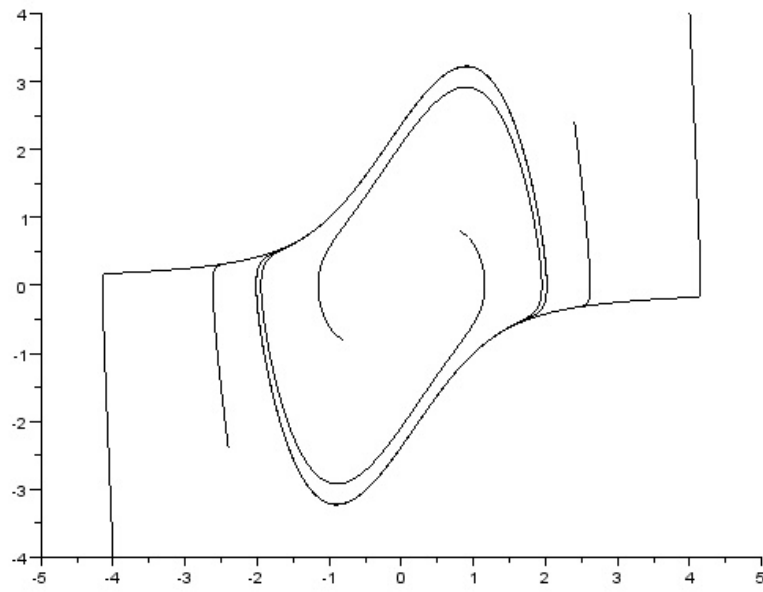


Figure 7.10: Lienard Systems



```

4           //x(dot);    x(2) means y.
5           //y(dot);    x(1) means x.;
6  endfunction
7  bound=[-4,-4,4,4];           //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them
    according to your needs.
8  nrect=12;           //increase it to get more number of
    curves, i.e. more information will be available
    .
9  set(gca(),"auto_clear","off")           //hold on
10 x=linspace(bound(1),bound(3),nrect);
11 y=linspace(bound(2),bound(4),nrect);
12 x0=[];
13
14 for i=1:12
15     x0=[x(i);y(i)];
16     t0=0;
17     t=0:0.01:3000;
18     xout=ode(x0,t0,t,linear741);
19     plot2d(xout(1,:),xout(2,:));
20 end
21     xtitle('Phase Portrait','x-axis ( x )','y-axis (
    y )')
```

---

### Scilab code Exa 7.5.1 Relaxation Oscillations

```

1 mew=10;
2 function xd=linear751(t,x)
3     F=(1/3)*(x(1)^3)-x(1);
4     xd(1)=mew*(x(2)-F);
5     xd(2)=- (1/mew)*x(1);
6           //x(dot);    x(2) means y.
7           //y(dot);    x(1) means x.;
```

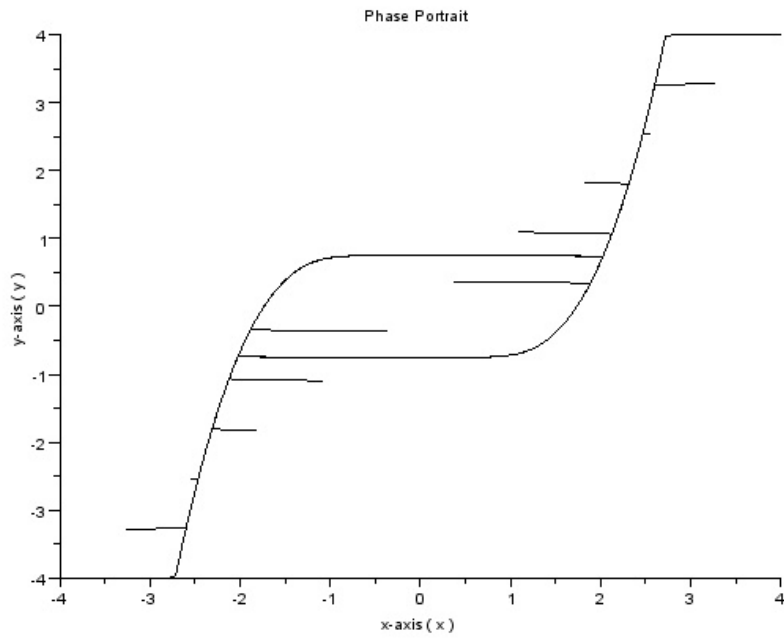


Figure 7.11: Relaxation Oscillations

```

8  endfunction
9  bound=[-4,-4,4,4];          //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them
    according to your needs.
10 nrect=12;                  //increase it to get more number of
    curves, i.e. more information will be available
    .
11 set(gca(),"auto_clear","off")      //hold on
12 x=linspace(bound(1),bound(3),nrect);
13 y=linspace(bound(2),bound(4),nrect);
14 x0=[];
15
16 for i=1:12
17     x0=[x(i);y(i)];
18     t0=0;
19     t=0:0.01:3000;
20     xout=ode(x0,t0,t,linear751);
21     plot2d(xout(1,:),xout(2,:));
22 end
23 xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')

```

---

# Chapter 8

## Bifurcations Revisited

Scilab code Exa 8.1.1 Saddle Node Transcritical Pitchfork Bifurcations

```
1 clear;
2 clear;
3 clc;
4 close;
5 set(gca(),"auto_clear","off")           //hold on
6
7 a=0.4;
8 b=0.8;
9 for x=0:0.1:3
10     y1=a*x;
11     y2=(x^2)/(b*(1+x^2));
12     plot2d(x,y1,style=-2)
13     plot2d(x,y2,style=-3)
14 end
15
16 // Classification of fixed points :
17
```

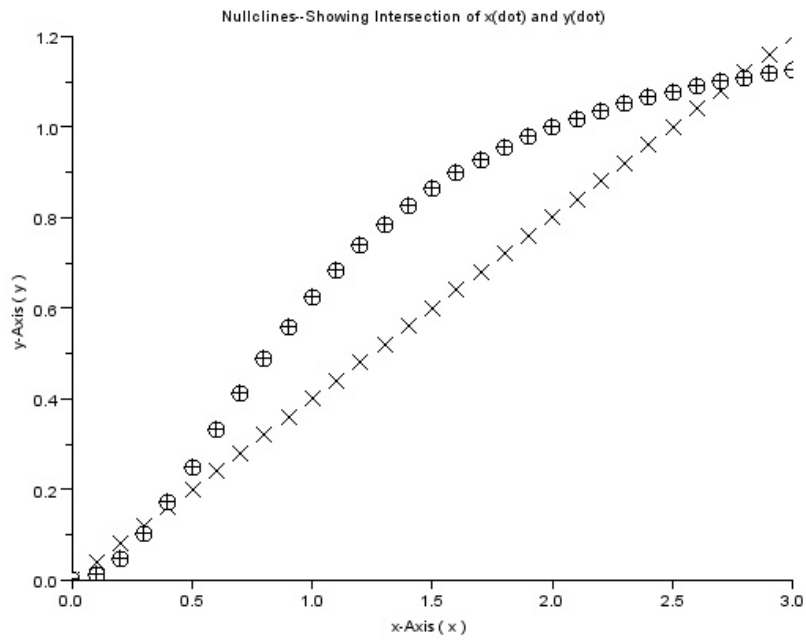


Figure 8.1: Saddle Node Transcritical Pitchfork Bifurcations

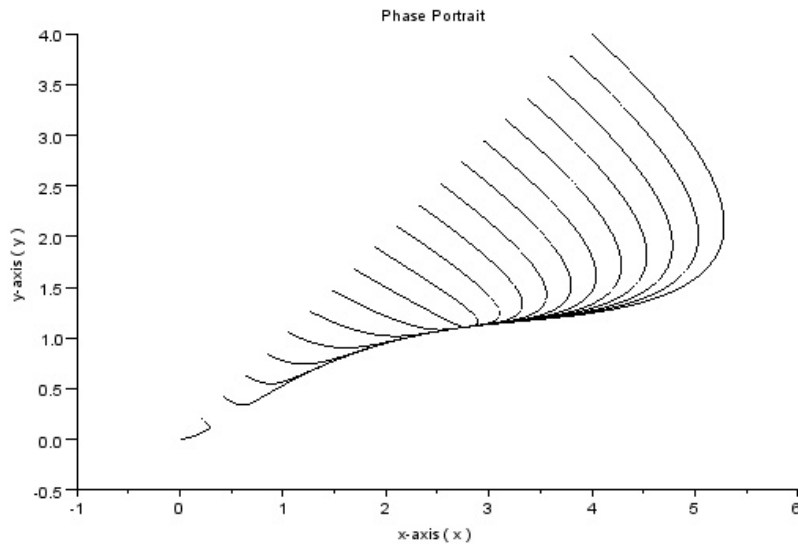


Figure 8.2: Saddle Node Transcritical Pitchfork Bifurcations

```

18 A1=[-a 1;0 -b]           //Jacobian at (0,0)
19 T=trace(A1)             //Trace of A
20 D=det(A1)               //Determinant of A
21
22 disp("Since , D>0, T<0 , origin is always a fixed
      point.")
23
24 //Now using the arguments given in book and the
      figure obtained through this example, we conclude
      :
25
26 disp("Middle Fixed Point lies between 0<x*<1, Thus
      is a Saddle Point.")
27 disp("The Thied fixed point is with x*>1, Thus
      always a stable node.")
28
29 xtitle("Nullclines —Showing Intersection of x(dot)
      and y(dot)", "x-Axis ( x )", "y-Axis ( y )")
30 figure

```

```

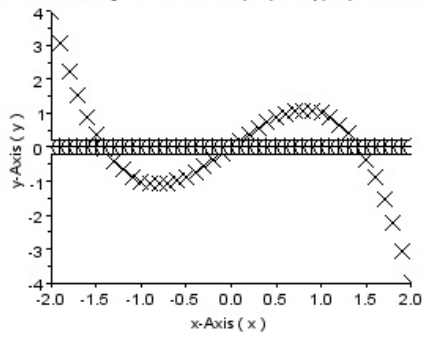
31 a=0.4;
32 b=0.8;
33 function xd=linear811(t,x)
34     xd(1)=-a*x(1)+x(2);
35     xd(2)=((x(1)^2)/(1+x(1)^2))-b*x(2);
36             //x(dot);    x(2) means y.
37             //y(dot);    x(1) means x.;
38 endfunction
39 bound=[0,0,4,4];           //Bounds of x-axis and y-
                             axis as [xmin ymin xmax ymax], change them
                             according to your needs.
40 nrect=20;                 //increase it to get more number of
                             curves, i.e. more information will be available
                             .
41 set(gca(),"auto_clear","off")           //hold on
42 x=linspace(bound(1),bound(3),nrect);
43 y=linspace(bound(2),bound(4),nrect);
44 x0=[];
45
46 for i=1:20
47     x0=[x(i);y(i)];
48     t0=0;
49     t=0:0.01:3000;
50     xout=ode(x0,t0,t,linear811);
51     plot2d(xout(1,:),xout(2,:));
52 end
53 xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
        )')

```

---

Scilab code Exa 8.1.2 Saddle Node Transcritical Pitchfork Bifurcations

Nullclines-Showing Intersection of  $x(\dot{\phantom{x}})$  and  $y(\dot{\phantom{y}})$  for  $\mu > 0$



Nullclines-Showing Intersection of  $x(\dot{\phantom{x}})$  and  $y(\dot{\phantom{y}})$  for  $\mu < 0$

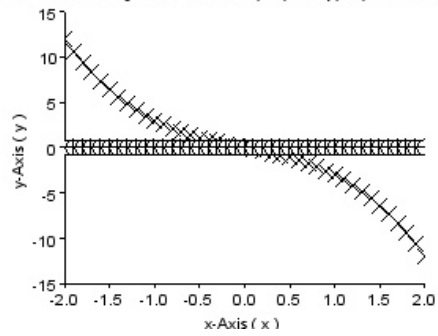


Figure 8.3: Saddle Node Transcritical Pitchfork Bifurcations



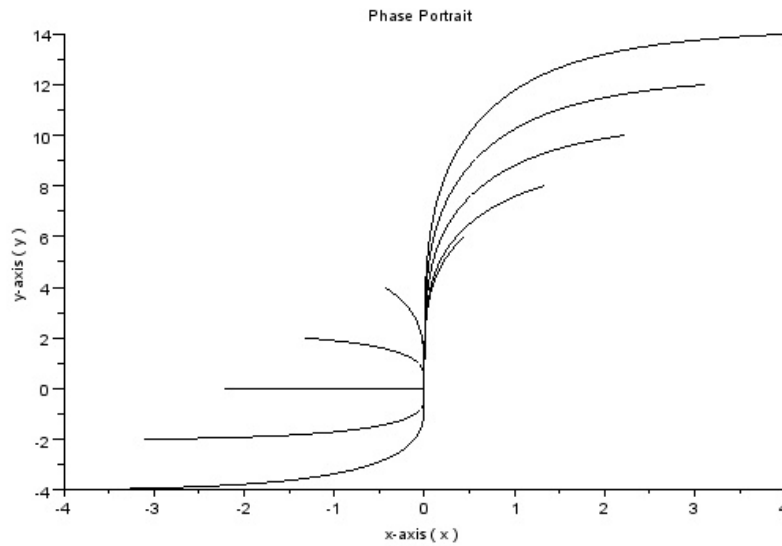


Figure 8.4: Saddle Node Transcritical Pitchfork Bifurcations

```

1 clear;
2 clear;
3 clc;
4 close;
5 set(gca()," auto_clear"," off")           //hold on
6
7 //f = x(dot)
8 //g = y(dot)
9 //Obtain Jacobian as :
10 //J=[df/dx  df/dy; dg/dx  dg/dy]
11 //Thus, J=[mew-3*x^2  0; 0  -1]
12 mew=2;
13 subplot(221)
14 for x=-2:0.1:2
15     y1=(mew*x)-x^3;
16     y2=0;
17     plot2d(x,y1,style=-2)
18     plot2d(x,y2,style=-3)
19 end

```

```

20 xtitle(" Nullclines —Showing Intersection of x(dot)
      and y(dot) for mew > 0", "x-Axis ( x )", "y-Axis (
      y )")
21 //Stabilities for mew >0
22 A1=[mew 0;0 -1]           //Jacobian at (0,0)
23 T1=trace(A1)
24 D1=det(A1)
25 A2=[-2*mew 0;0 -1]       //Jacobian at (+sqrt(mew)
      ,0) and (-sqrt(mew),0)
26 T2=trace(A2)
27 D2=det(A2)
28 Det = (T2^2) - 4*D2
29
30 disp(" Since D1<0, Thus (0,0) is a Saddle Point.")
31 disp(" Since T2<0, D2>0 and Det>0, thus (+sqrt(mew)
      ,0) and (-sqrt(mew),0) are stable points.")
32 subplot(224)
33 mew=-2;
34 for x=-2:0.1:2
35     y1=(mew*x)-x^3;
36     y2=0;
37     plot2d(x,y1,style=-2)
38     plot2d(x,y2,style=-3)
39 end
40 xtitle(" Nullclines —Showing Intersection of x(dot)
      and y(dot) for mew < 0", "x-Axis ( x )", "y-Axis (
      y )")
41 // Classification of fixed points :
42
43 A3=[mew 0;0 -1]           //Jacobian at (0,0)
44 T3=trace(A3)             //Trace of A
45 D3=det(A3)              //Determinant of A
46 Det3=T3^2 - 4*D3
47 disp(" Since , D>0, T<0 , and Det3>0 origin is a stable
      fixed point.")
48
49
50 //Note Stabilities can be deduced from the figures

```

```

    also as done in previous chapters.
51 figure
52 mew=-4;
53 function xd=linear812(t,x)
54     xd(1)=mew*x(1)-(x(1)^3);
55     xd(2)=-x(2);
56         //x(dot);    x(2) means y.
57         //y(dot);    x(1) means x.;
58 endfunction
59 bound=[-4,-4,4,14];           //Bounds of x-axis and
    y-axis as [xmin ymin xmax ymax], change them
    according to your needs.
60 nrect=10;           //increase it to get more number of
    curves, i.e. more information will be available
    .
61 set(gca(),"auto_clear","off")           //hold on
62 x=linspace(bound(1),bound(3),nrect);
63 y=linspace(bound(2),bound(4),nrect);
64 x0=[];
65
66 for i=1:10
67     x0=[x(i);y(i)];
68     t0=0;
69     t=0:0.01:3000;
70     xout=ode(x0,t0,t,linear812);
71     plot2d(xout(1,:),xout(2,:));
72 end
73 xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')
```

---

### Scilab code Exa 8.1.3 Saddle Node Transcritical Pitchfork Bifurcations

```
1 mew=-2.1;
```

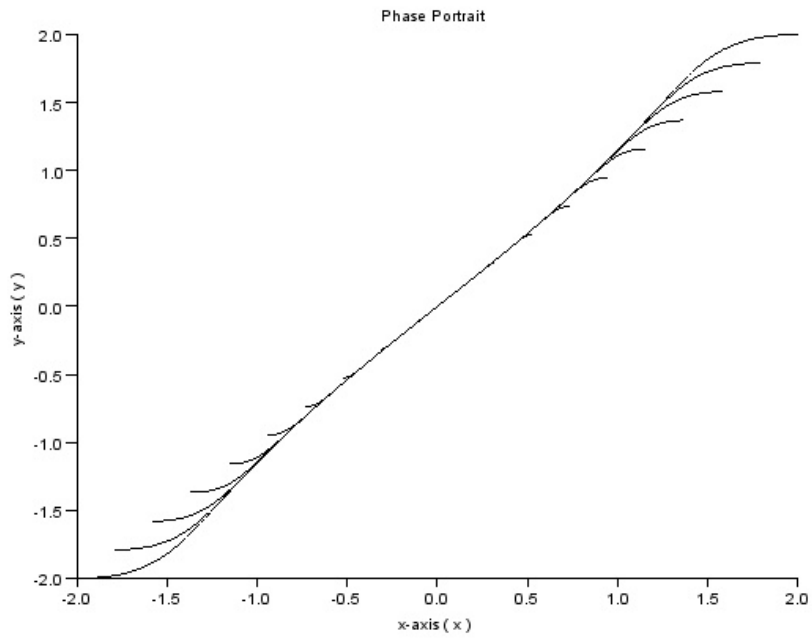


Figure 8.5: Saddle Node Transcritical Pitchfork Bifurcations

```

2  function xd=linear813(t,x)
3      xd(1)=mew*x(1)+x(2)+sin(x(1));
4      xd(2)=x(1)-x(2);
5          //x(dot);    x(2) means y.
6          //y(dot);    x(1) means x.;
7  endfunction
8  bound=[-2,-2,2,2];          //Bounds of x-axis and y
    -axis as [xmin ymin xmax ymax], change them
    according to your needs.
9  nrect=20;          //increase it to get more number of
    curves, i.e. more information will be available
    .
10 set(gca(),"auto_clear","off")          //hold on
11 x=linspace(bound(1),bound(3),nrect);
12 y=linspace(bound(2),bound(4),nrect);
13 x0=[];
14
15 for i=1:20
16     x0=[x(i);y(i)];
17     t0=0;
18     t=0:0.01:3000;
19     xout=ode(x0,t0,t,linear813);
20     plot2d(xout(1,:),xout(2,:));
21 end
22 xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')

```

---

### Scilab code Exa 8.2.1 Hopf Bifurcations

```

1 mew=-0.2;
2  function xd=linear821(t,x)
3      xd(1)=mew*x(1)-x(2)+x(1)*x(2)^2;
4      xd(2)=x(1)+mew*x(2)+x(2)^3;

```

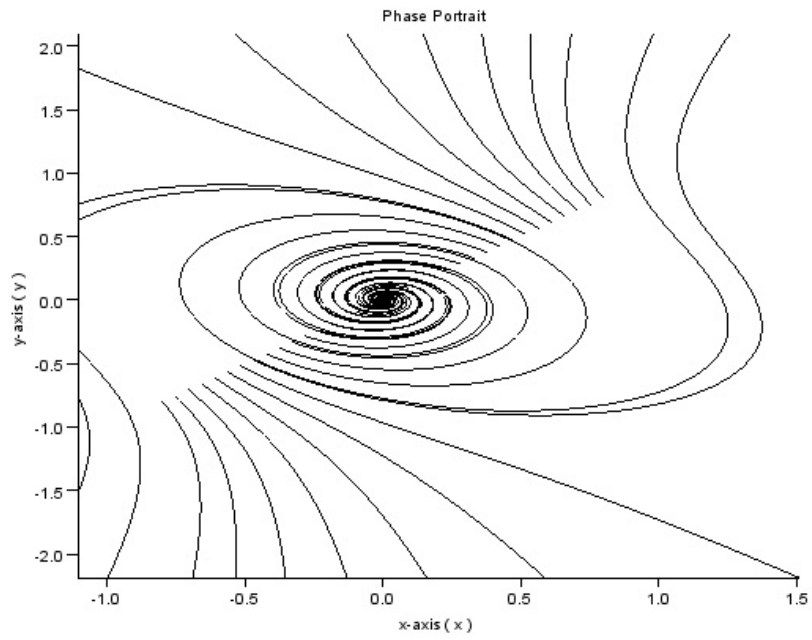


Figure 8.6: Hopf Bifurcations

```

5           //x(dot);    x(2) means y.
6           //y(dot);    x(1) means x.;
7   endfunction
8   bound=[-0.8,-0.8,0.8,0.8];           //Bounds of x-
    axis and y-axis as [xmin ymin xmax ymax],
    change them according to your needs.
9   nrect=35;           //increase it to get more number of
    curves, i.e. more information will be available
    .
10  set(gca(),"auto_clear","off")           //hold on
11  x=linspace(bound(1),bound(3),nrect);
12  y=linspace(bound(2),bound(4),nrect);
13  x0=[];
14
15  for i=1:35
16      x0=[x(i);y(i)];
17      t0=0;
18      t=0:0.01:3000;
19      xout=ode(x0,t0,t,linear821);
20      plot2d(xout(1,:),xout(2,:));
21  end
22  xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')
```

---

### Scilab code Exa 8.3.1 Oscillating Chemical Reactions

```

1  a=10;
2  b=3;
3  function xd=linear831(t,x)
4      xd(1)=a-x(1)-((4*x(1)*x(2))/(1+x(1)^2));
5      xd(2)=(b*x(1))*(1-(x(2)/(1+x(1)^2)));
6           //x(dot);    x(2) means y.
7           //y(dot);    x(1) means x.;
```

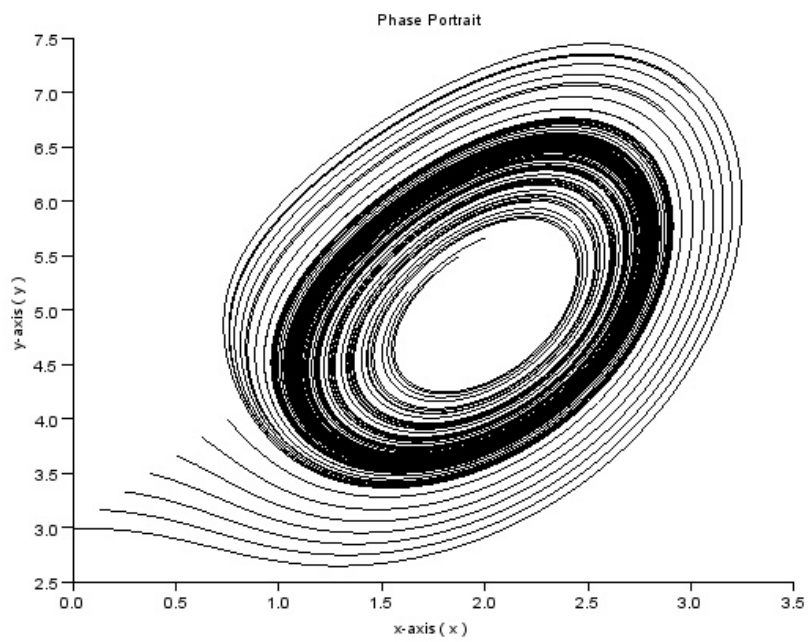


Figure 8.7: Oscillating Chemical Reactions



```

8  endfunction
9  bound=[0,3,3,7];           //Bounds of x-axis and y-
    axis as [xmin ymin xmax ymax], change them
    according to your needs.
10 nrect=25;                 //increase it to get more number of
    curves, i.e. more information will be available
    .
11 set(gca(),"auto_clear","off")           //hold on
12 x=linspace(bound(1),bound(3),nrect);
13 y=linspace(bound(2),bound(4),nrect);
14 x0=[];
15
16 for i=1:25
17     x0=[x(i);y(i)];
18     t0=0;
19     t=0:0.01:3000;
20     xout=ode(x0,t0,t,linear831);
21     plot2d(xout(1,:),xout(2,:));
22 end
23 xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
    )')

```

---

### Scilab code Exa 8.3.2 Oscillating Chemical Reactions

```

1  a=10;
2  b=4;
3  function xd=linear832(t,x)
4      xd(1)=a-x(1)-((4*x(1)*x(2))/(1+x(1)^2));
5      xd(2)=(b*x(1))*(1-(x(2)/(1+x(1)^2)));
6          //x(dot);    x(2) means y.
7          //y(dot);    x(1) means x.;

```

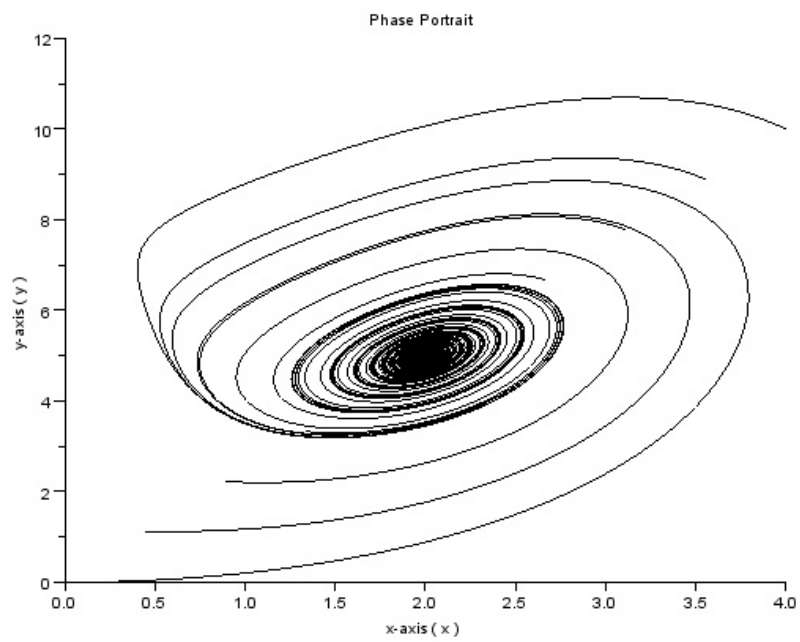


Figure 8.8: Oscillating Chemical Reactions

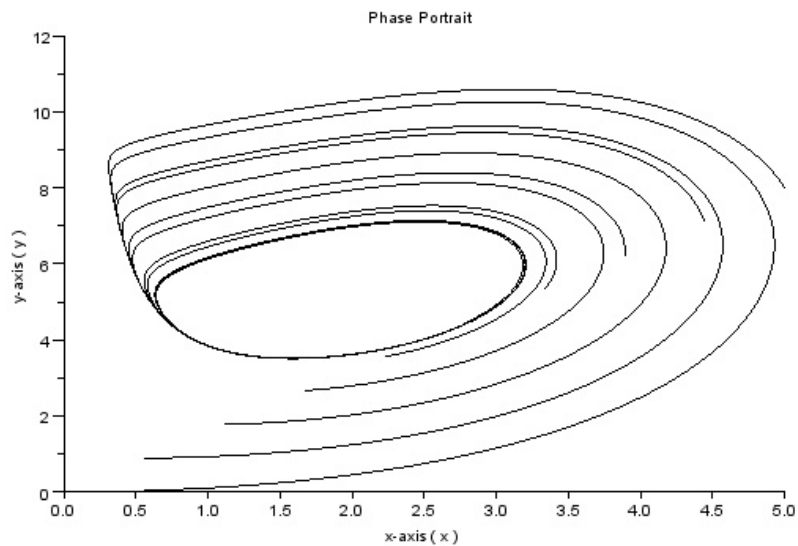


Figure 8.9: Oscillating Chemical Reactions

```

8  endfunction
9  bound=[0,0,4,10];          //Bounds of x-axis and y-
    axis as [xmin ymin xmax ymax], change them
    according to your needs.
10 nrect=10;                //increase it to get more number of
    curves, i.e. more information will be available
    .
11 set(gca(),"auto_clear","off")    //hold on
12 x=linspace(bound(1),bound(3),nrect);
13 y=linspace(bound(2),bound(4),nrect);
14 x0=[];
15
16 for i=1:10
17     x0=[x(i);y(i)];
18     t0=0;
19     t=0:0.01:3000;
20     xout=ode(x0,t0,t,linear832);
21     plot2d(xout(1,:),xout(2,:));
22 end

```

```

23  xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
      )')
24  figure
25  a=10;
26  b=2;
27  function xd=linear8322(t,x)
28      xd(1)=a-x(1)-((4*x(1)*x(2))/(1+x(1)^2));
29      xd(2)=(b*x(1))*(1-(x(2)/(1+x(1)^2)));
30          //x(dot);    x(2) means y.
31          //y(dot);    x(1) means x.;
32  endfunction
33  bound=[0,0,5,8];          //Bounds of x-axis and y-
      axis as [xmin ymin xmax ymax], change them
      according to your needs.
34  nrect=10;                //increase it to get more number of
      curves, i.e. more information will be available
      .
35  set(gca(),'auto_clear','off')          //hold on
36  x=linspace(bound(1),bound(3),nrect);
37  y=linspace(bound(2),bound(4),nrect);
38  x0=[];
39
40  for i=1:10
41      x0=[x(i);y(i)];
42      t0=0;
43      t=0:0.01:3000;
44      xout=ode(x0,t0,t,linear8322);
45      plot2d(xout(1,:),xout(2,:));
46  end
47  xtitle('Phase Portrait','x-axis ( x )','y-axis ( y
      )')

```

---

# Chapter 10

## One Dimensional Maps

Scilab code Exa 10.1.1 Fixed Points and Cobwebs

```
1 //Example 10.1.1 Page 350
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 x=poly(0,"x");
10 f = (x^2)-x; //Defining
   Polynomial—>  $x(\text{dot})=x^2 - 1$ . Let this be  $f(x)$ 
11 disp("Fixed Points are :")
12 y = roots(f)
13
14 lambda1=evstr(2*y(1))
15 lambda2=evstr(2*y(2))
16
17 // if lambda1<1 then
18 //   disp(y(1))
19 //disp(" Stable.")
```

```

20 //elseif lambda1>1
21     // disp(y(1))
22     //disp(" Unstable.")
23 //else
24 //     disp(" Unconclusive since lambda=1.")
25 //end
26
27 //if lambda2<1 then
28 //     disp(y(2))
29 //     disp(" Stable.")
30 //elseif lambda2>1
31 //     disp(y(2))
32 //     disp(" Unstable.")
33 //else
34 //     disp(" Unconclusive since lambda=1.")
35 //end
36
37 disp(" Since lambda1=0<1, Thus it is stable.")
38 disp(" Since lambda2=2>1 Thus it is unstable.")
39
40 //End of Example.

```

---

### Scilab code Exa 10.1.2 Fixed Points and Cobwebs

```

1 //Example 10.1.2 Page 351
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;

```

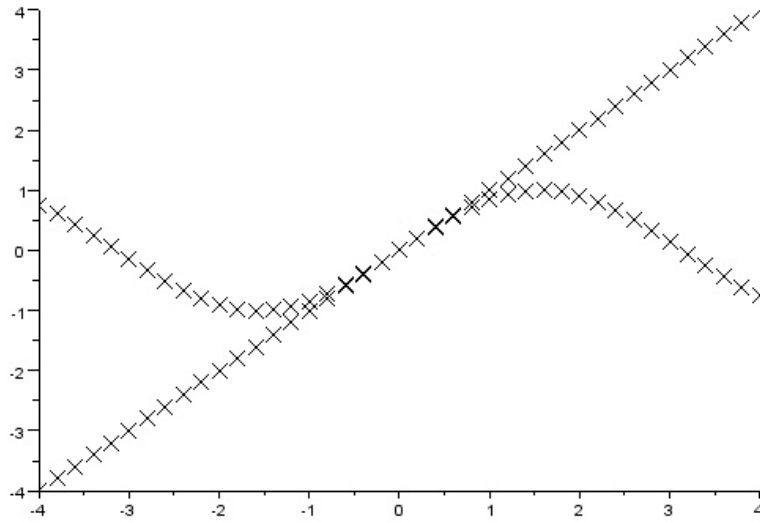


Figure 10.1: Fixed Points and Cobwebs

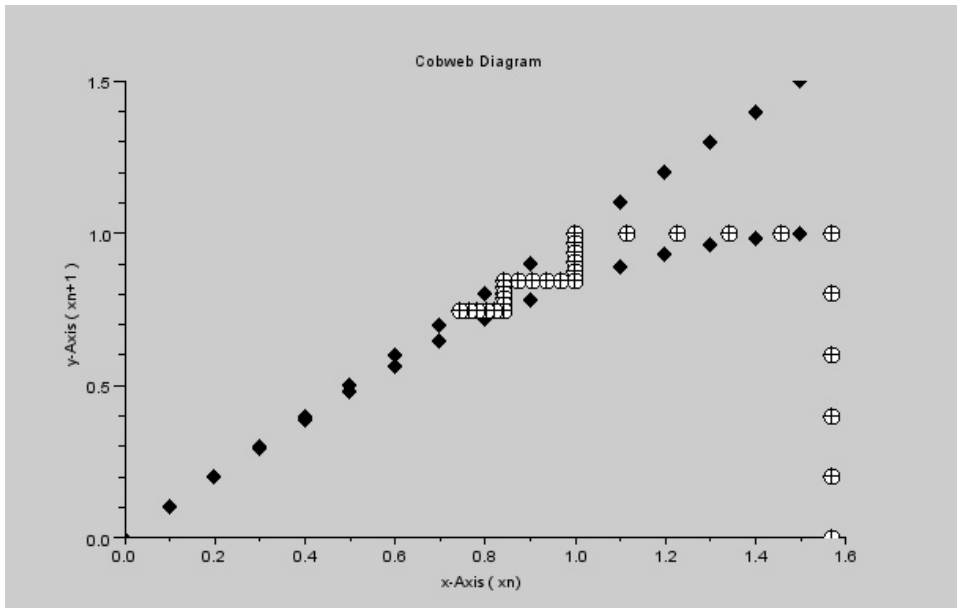


Figure 10.2: Fixed Points and Cobwebs

```

7 close;
8 set(gca(),"auto_clear","off")           //hold on
9
10 for x=-4:0.2:4
11     y1=sin(x);
12     y2=x;
13     plot2d(x,y1,style=-2)
14     plot2d(x,y2,style=-2)
15 end
16
17 disp("By zooming the above graph we observe that
        only fixed point is x=0.")
18
19 lambda=cos(0)           //f'(x*)=cos(0)
20
21 disp("Since, lambda=1; therefore we can not say
        about stability of x*=0.")
22 disp("So, see stability by cobwebs diagram, as shown
        in book.")
23
24 //Cobweb Diagram :-
25
26 figure
27 for x=0:0.1:(%pi/2)
28     y1=sin(x);
29     y2=x;
30     plot2d(x,y1,style=-4)
31     plot2d(x,y2,style=-4)
32 end
33
34 x0=%pi/2;
35 for y=0:0.2:sin(x0)
36     x=x0;
37     plot2d(x,y,style=-3)
38 end
39 dx1=(sin(x0)-x0)/5;
40 for x=x0:dx1:sin(x0)
41     y=sin(x0);

```



```

42     plot2d(x,y,style=-3)
43 end
44 dy1=(sin(sin(x0))-sin(x0))/5;
45 for y=sin(x0):dy1:sin(sin(x0))
46     x=sin(x0);
47     plot2d(x,y,style=-3)
48 end
49 dx2=(sin(sin(x0))-sin(x0))/5;
50 for x=sin(x0):dx2:sin(sin(x0))
51     y=sin(sin(x0));
52     plot2d(x,y,style=-3)
53 end
54 x=sin(sin(x0));
55 dy2=(sin(x)-x)/5;
56 for y=x:dy2:sin(x)
57     x=sin(sin(x0));
58     plot2d(x,y,style=-3)
59 end
60 y=sin(sin(sin(x0)));
61 dx3=-((sin(sin(x0))-y)/5);
62 for x=sin(sin(x0)):dx3:y
63     //y=sin(x);
64
65     plot2d(x,y,style=-3)
66 end
67 xtitle("Cobweb Diagram", "x-Axis ( xn)", "y-Axis ( xn
+1 )")
68 //End of Example.

```

---

### Scilab code Exa 10.3.1 Logistic Map

```

1 //Example 10.3.1 Page 357
2 //Non-Linear Dynamics and Chaos, First Indian

```

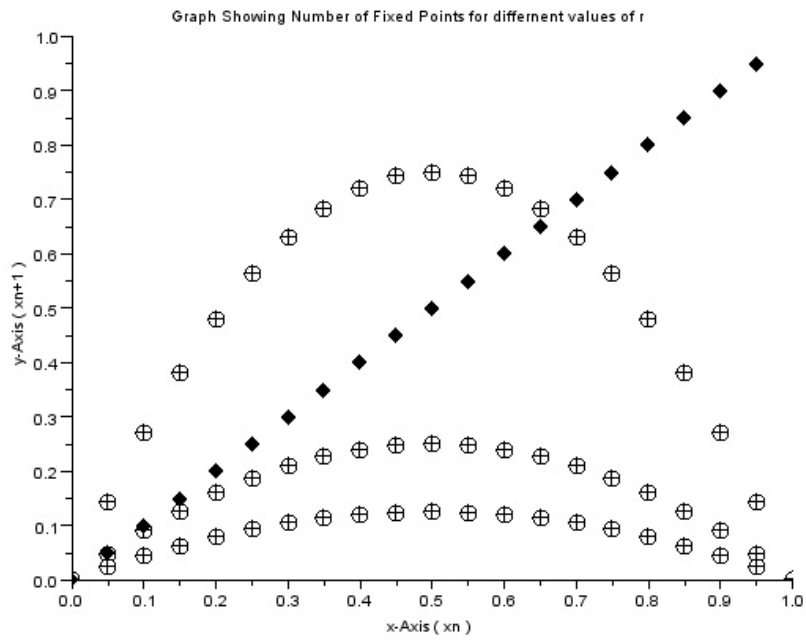


Figure 10.3: Logistic Map

```

    Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8 set(gca()," auto_clear"," off")           //hold on
9
10 //Taking r=2;
11 r=2;
12 x=poly(0,"x");
13 f = x-2*(x^2);                               //Defining
    Polynomial—>  $f(x^*)-x^* = 2*x(1-x)-x$ . Let this be
     $f(x)$ 
14 disp("Fixed Points are :")
15 y = roots(f)
16
17 disp("The fixed point  $x^*=1-(1/r)$  does not exists for
     $r<1$ , Since  $x(n+1)<0$  and population cannot be
    negative.")
18
19 lambda1=r-2*r*y(1)           //f'(x*) = r-2rx*
20 lambda2=r-2*r*y(2)
21
22 disp("Since , lambda1=2>1, thus orign is Unstable.")
23 disp("Since , lambda2=0<1, thus  $x^*=1-(1/r)$  is Stable.
    ")
24
25 //Number of points graphically :
26
27 r1=3;           //r>1
28 r2=1;           //r=1, tangential case
29 r3=0.5;        //r<1
30
31 for xn=0:0.05:1
32     xn_one=r1*xn*(1-xn);
33     plot2d(xn,xn_one,style=-3)
34     xn_one=r2*xn*(1-xn);

```

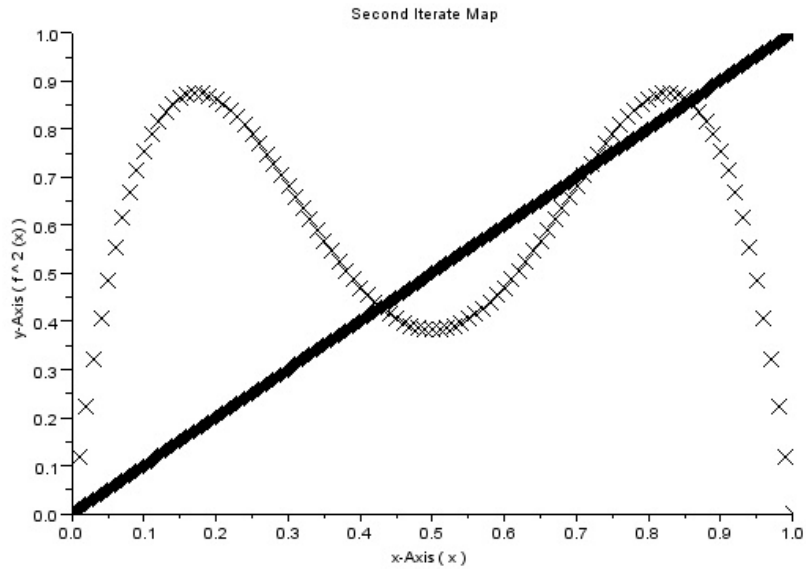


Figure 10.4: Logistic Map

```

35     plot2d(xn,xn_one,style=-3)
36     xn_one=r3*xn*(1-xn);
37     plot2d(xn,xn_one,style=-3)
38     y=xn;          // to draw y=x line
39     plot2d(xn,y,style=-4)
40 end
41 xtitle("Graph Showing Number of Fixed Points for
         different values of r","x-Axis ( xn )","y-Axis (
         xn+1 )")
42
43 //Similarly, check for Stability by changing r.
44
45 //End of Example.

```

---

### Scilab code Exa 10.3.2 Logistic Map

```
1 //Example 10.3.2 Page 358
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off")           //hold on
9
10 //Taking r=3.5 such that r>3 as given in book.;
11 r=3.5;
12 for x=0:0.01:1
13     f2x=((r^2)*x)-(r*x)^2*(1-r*x+r*(x^2));
        //f2x=second-iterate map.
14     plot2d(x,f2x,style=-2)
15     y=x;           //to plot y=x line.
16     plot2d(x,y,style=-4)
17 end
18 xtitle("Second Iterate Map","x-Axis ( x )","y-Axis (
   f ^ 2 (x) )")
```

---

### Scilab code Exa 10.3.3 Logistic Map

```
1 //Example 10.3.2 Page 358
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off")           //hold on
```

```

9
10 r=0:0.1:5;
11 size1=length(r);
12 x=zeros(601,size1);
13 x(1,1:size1)=0.1;
14 for r1=1:size1
15     for n=1:600
16         x(n+1,r1)=r(r1)*x(n,r1)*(1-x(n,r1));
17     end
18 end
19
20 //z=zeros(301,size1);
21 for r1=1:size1
22     for n=300:600
23         //z(n-299,r1)=x(n+1,r1);
24         plot2d(r(r1),x(n+1),style=-2)
25     end
26 end
27
28 // End of Example

```

---

### Scilab code Exa 10.6.1 Universality and Experiments

```

1 //Example 10.6.1 Page 369
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8 set(gca(),"auto_clear","off") //hold on
9

```

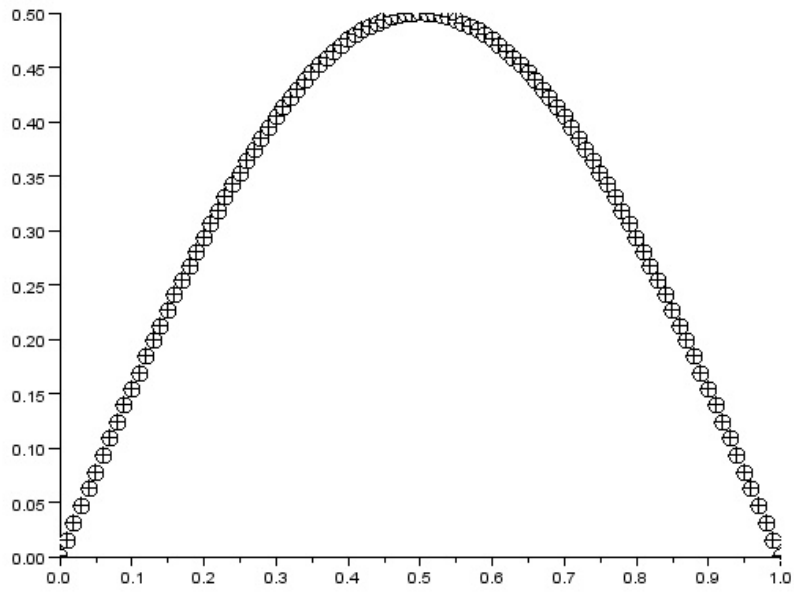


Figure 10.5: Universality and Experiments

```

10 //Sine Map
11 r=0.5;
12 for xn=0:0.01:1
13     xn_one=r*sin(%pi*xn);
14     plot2d(xn,xn_one,style=-3)
15 end

```

---

### Scilab code Exa 10.7.2 Renormalization

```

1 //Example 10.7.2 Page 386
2 //Non-Linear Dynamics and Chaos, First Indian
  Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 x=poly(0,"x");
10 f = (x^2)+4*x-2; //Defining
  Polynomial—>  $x(\dot{)}=x^2 -1$ . Let this be  $f(x)$ 
11 disp("Fixed Points are :")
12 y = roots(f);
13
14 //  $-2+\sqrt{6}=0.4494897$ 
15
16 // As seen earlier period 2 cycle occurs at  $r=3$ .
17
18 //Hence,
19
20 r2=y(1)+3
21
22
23 //End of Example.

```

---



### Scilab code Exa 10.7.3 Renormalization

```
1 //Example 10.7.2 Page 386
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 x=poly(0,"x");
10 f = (x^2)+3*x-2; //Defining
   Polynomial—>  $x(\dot{)}=x^2 -1$ . Let this be  $f(x)$ 
11 disp("Fixed Points are :")
12 y = roots(f)
13
14 // As seen earlier period 2 cycle occurs at  $r=3$ .
15
16 //Hence ,
17
18 r2=y(1)+3
19
20
21 //End of Example.
```

---

# Chapter 11

## Fractals

Scilab code Exa 11.3.1 Dimension of Self Similar Fractals

```
1 //Example 11.3.1 Page 407
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 // m = number of copies.
10 // r = scale factor.
11 m=2;
12 r=3;
13
14 d=log(m)/log(r)
15
16 //End of Example.
```

---

Scilab code Exa 11.3.2 Dimension of Self Similar Fractals

```

1 //Example 11.3.2 Page 407
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 // m = number of copies.
10 // r = scale factor.
11 m=4;
12 r=3;
13
14 d=log(m)/log(r)
15
16 //End of Example.

```

---

### Scilab code Exa 11.3.3 Dimension of Self Similar Fractals

```

1 //Example 11.3.3 Page 408
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 // m = number of copies.
10 // r = scale factor.
11 m=3;
12 r=5;
13
14 d=log(m)/log(r)

```

```
15
16 //End of Example.
```

---

#### Scilab code Exa 11.4.1 Box Dimension

```
1 //Example 11.4.1 Page 409
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 // N(E) = minimum number of D-dimensional cubes of
   side E needed to cover S.
10
11 n=poly(0,"n") //Defining
   polynomial in "n"
12 Num = (n*log(2)) //Num. =
   Numerator
13 Den = (n*log(3)) //Den. =
   Denominator
14 disp("The box dimension is :")
15
16 d=(Num)/(Den) //d = Box
   Dimension.
17
18
19 //End of Example.
```

---

#### Scilab code Exa 11.4.2 Box Dimension

```

1 //Example 11.4.2 Page 410
2 //Non-Linear Dynamics and Chaos, First Indian
   Edition Print 2007
3 //Steven H. Strogatz
4 clear;
5 clear;
6 clc;
7 close;
8
9 // N(E) = minimum number of D-dimensional cubes of
   side E needed to cover S.
10
11 n=poly(0,"n") // Defining
   polynomial in "n"
12 Num = (n*log(8)) //Num. =
   Numerator
13 Den = (n*log(3)) //Den. =
   Denominator
14 disp("The box dimension is :")
15
16 d=(Num)/(Den) //d = Box
   Dimension.
17
18
19 //End of Example.

```

---

# Appendix

## Scilab code AP 1 Drawing Circle

```
1 function [] = circle(centre,radius,NOP)
2
3     THETA=linspace(0,2*pi,NOP);
4     RHO=ones(1,NOP)*radius;
5     z=tan(THETA);
6     x=sqrt(((RHO).^2)./(1+(z.^2)));
7     y=x.*z;
8     x=x+centre(1);
9     y=y+centre(2);
10    plot2d(x,y,style=-2);
11    set(gca(),"auto_clear","off") //hold on
12    plot2d(-(x)+2*(centre(1)),y,style=-2);
13
14
15 //////////////// End of Function circle
```

---