Chaotic Dynamics: Numerical Simulations

Paulsamy Muruganandam

Department of Physics
Bharathidasan University
Tiruchirappalli – 620024

E-mail: anand@bdu.ac.in

- What is chaos?
- Chaos in prototype models
- Numerical methods
- Simulations using Python

Chaos

Bounded aperiodic motion in deterministic nonlinear dynamical systems with sensitive dependence on initial conditions

Chaos

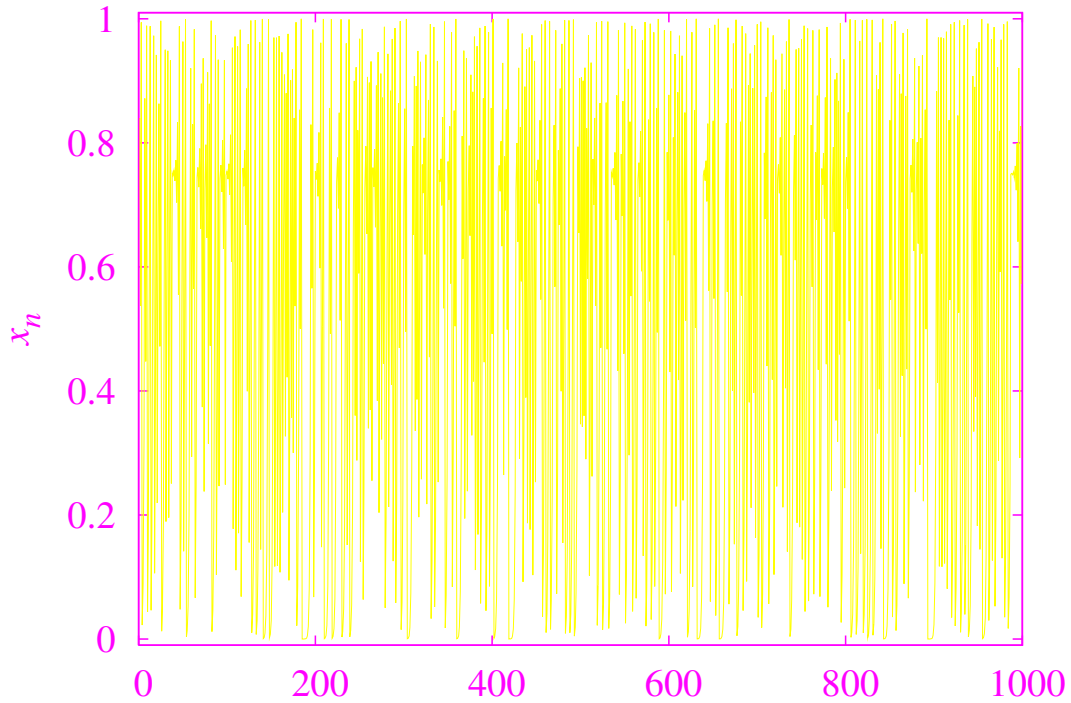
Bounded aperiodic motion in deterministic nonlinear dynamical systems with sensitive dependence on initial conditions

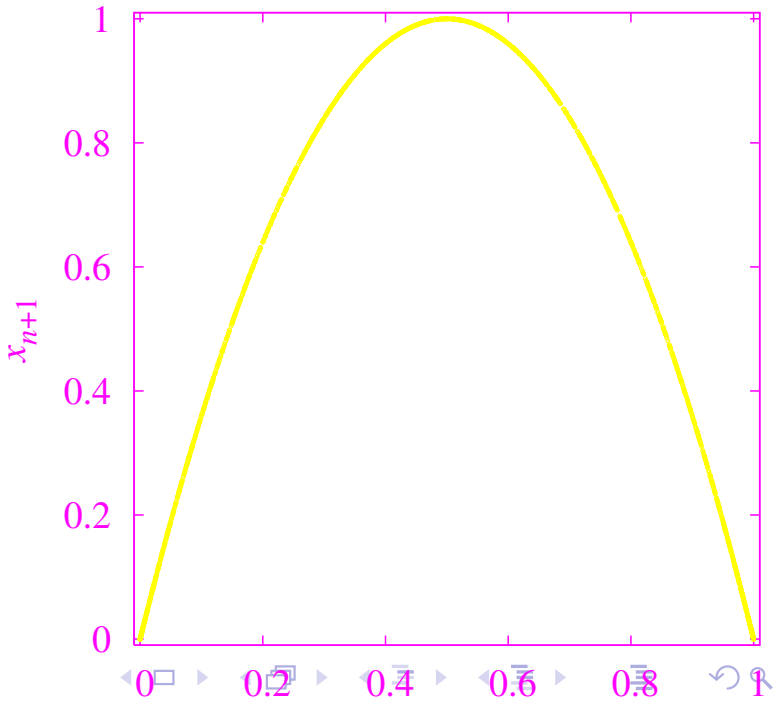
Characteristic features

- unpredictability (long time)
- neighbouring trajectories diverges exponentially (positive Lyapunov exponent)
- broad band (noise like) spectrum
- strange attractor and fractal dimensions

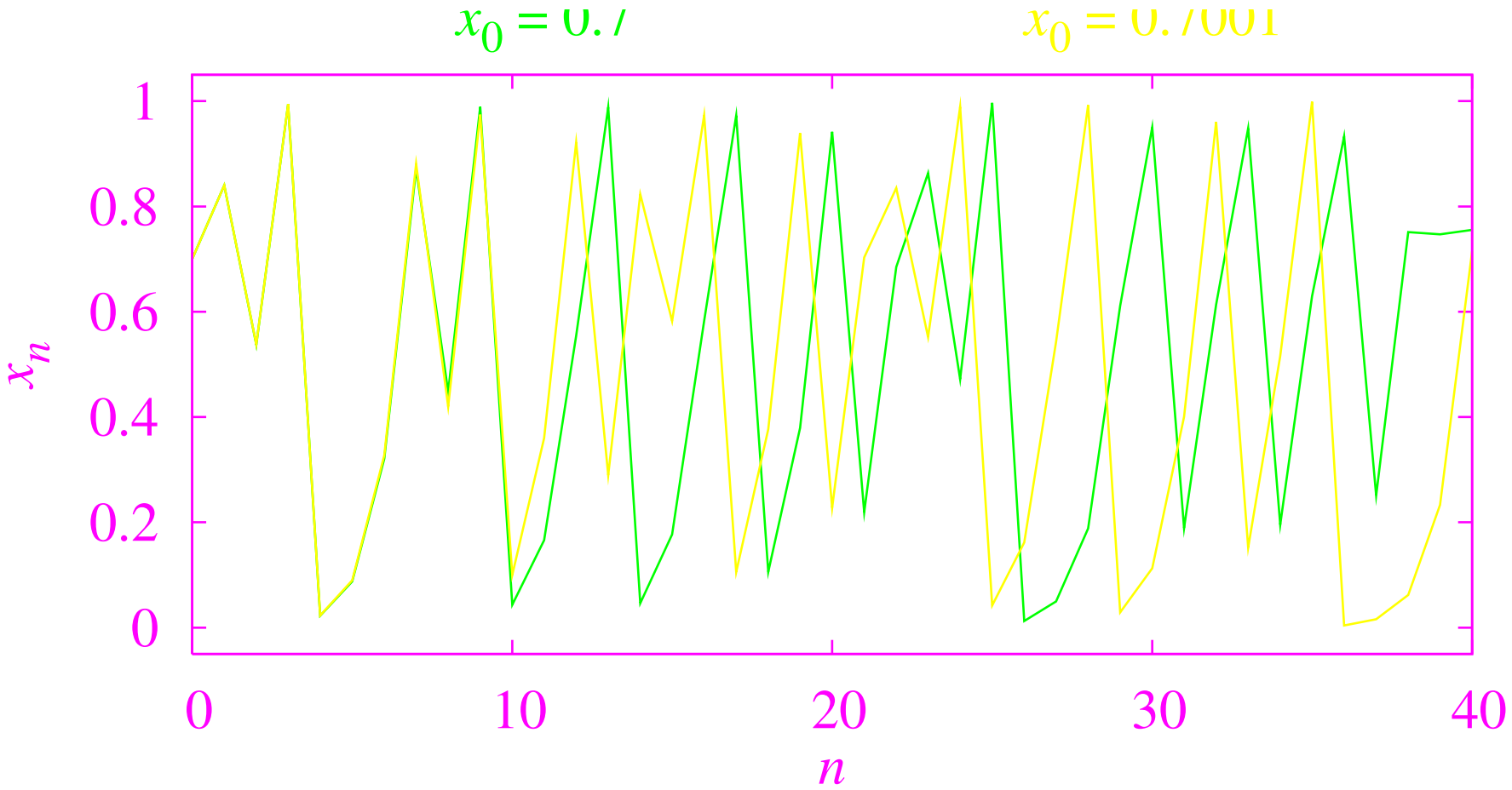
Logistic map

$$x_{n+1} = a x_n (1 - x_n), \quad x \in (0, 1), \quad a \in (0, 4)$$





Logistic map



Sensitive dependence on initial conditions

Why numerical solutions?

- In most of the occasions nonlinear differential equations (describing dynamical systems) – may not possess closed solutions (or) not exactly solvable.
- Local stability analysis can be done to understand the local dynamics to some extent.
- Numerical solutions becomes essential in order to study the complete dynamics (For eg. chaos and other related phenomena).

ODEs

$$\frac{dx_i}{dt} = f_i(x_1, x_2, x_3, \cdots, x_n, t), \quad i = 1, 2, \ldots, n \tag{1}$$

Any n-th order ODE of the form

$$\frac{d^n x}{dt^n} + a_1(t)\frac{d^{n-1}x}{dt^{n-1}} + \cdots + a_{n-1}(t)\frac{dx}{dt} + a_n(t)x = 0, \tag{2}$$

can always be written in the form (1) as $x_1 = x$,

$$\begin{aligned}
\dot{x}_1 &= x_2, \\
\dot{x}_2 &= x_3, \\
&\vdots \\
\dot{x}_n &= -a_1(t)x_{n-1} - \cdots - a_{n-1}(t)x_2 - a_n(t)x_1
\end{aligned} \tag{3}$$

Euler method

- Simple technique for handling first order initial value problems.
- Basic explicit method for solving ordinary differential equations.
- For simplicity, consider first order ODE $\dot{x} = f(x,t)$ with initial condition: $x(t_0) = x_0$

$$x_{n+1} = x_n + hf(x_n, t_n), \tag{4}$$

where $t_n = t_0 + nh$, h time step, and $x_n = x(t_n)$.

Forth order Runge-Kutta method

- Most commonly used method for solving ordinary differential equations – often referred to as "RK4".
- Consider the first order ODE $\dot{x} = f(x,t)$ with initial condition: $x(t_0) = x_0$

$$x_{n+1} = x_n + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right), \tag{5}$$
$$t_{n+1} = t_n + h,$$

where

$$k_1 = h\, f(x_n, t_n), \qquad k_2 = h\, f\left(x_n + \frac{k_1}{2}, t_n + \frac{h}{2}\right),$$
$$k_3 = h\, f\left(x_n + \frac{k_2}{2}, t_n + \frac{h}{2}\right), \qquad k_3 = h\, f\left(x_n + k_3, t_n + h\right). \tag{6}$$

- Euler method

$$E(x(b), h) = O(h), \quad t \in (a, b)$$

- Forth order Runge-Kutta method

$$E(x(b), h) = O(h^4), \quad t \in (a, b)$$

Python

- an interpreted, object-oriented, high-level programming language
- Simple and easy to learn syntax
- Scripting and non-scripting contexts

Python Modules

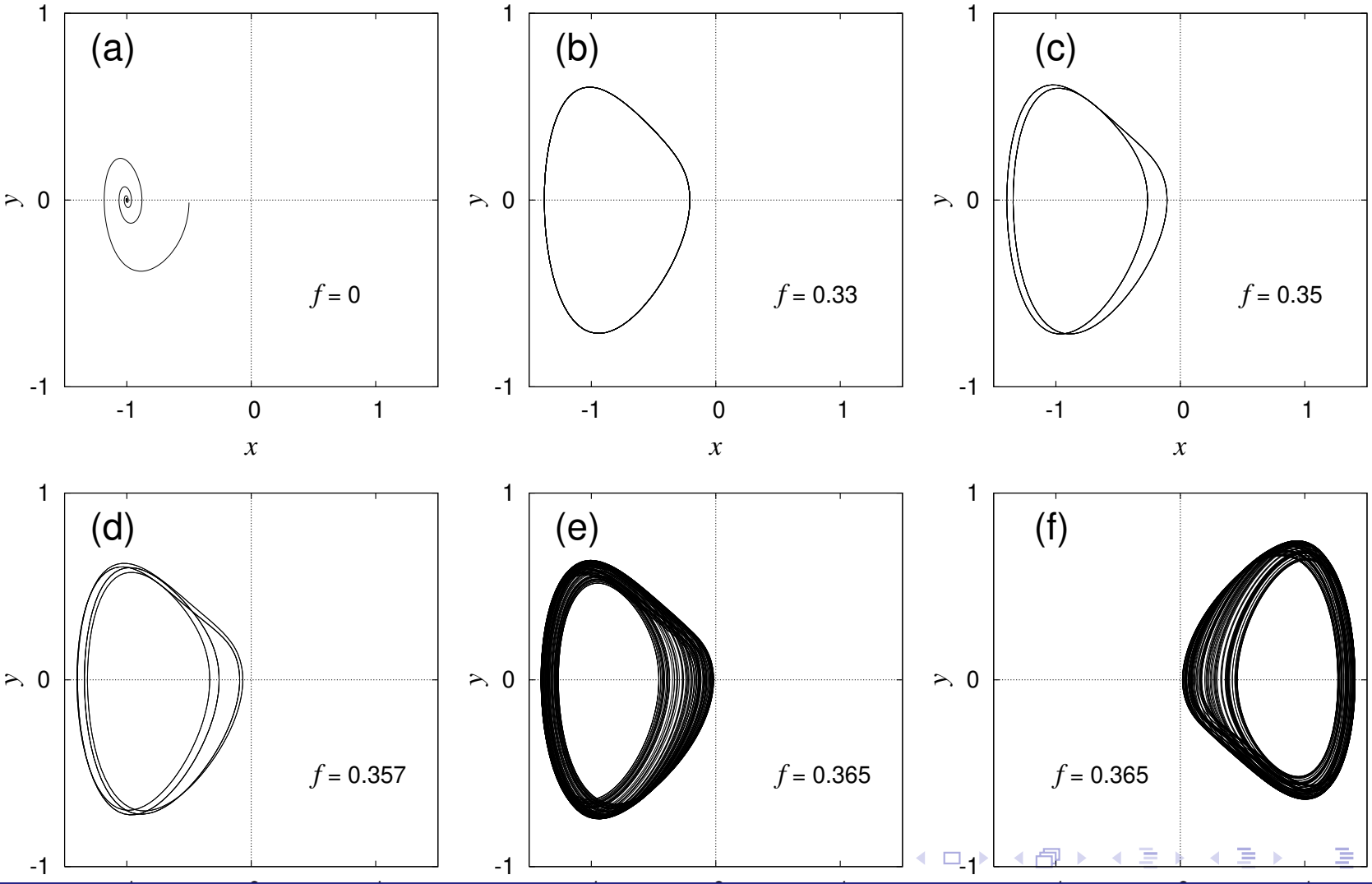
- SciPy – an open source library of algorithms and mathematical tools
- NumPy – an extension to Python – support of large multi-dimensional arrays and matrices – library of high-level mathematical function – free alternative to MATLAB
- Matplotlib – plotting library for the Python programming language and its NumPy numerical mathematics extensions

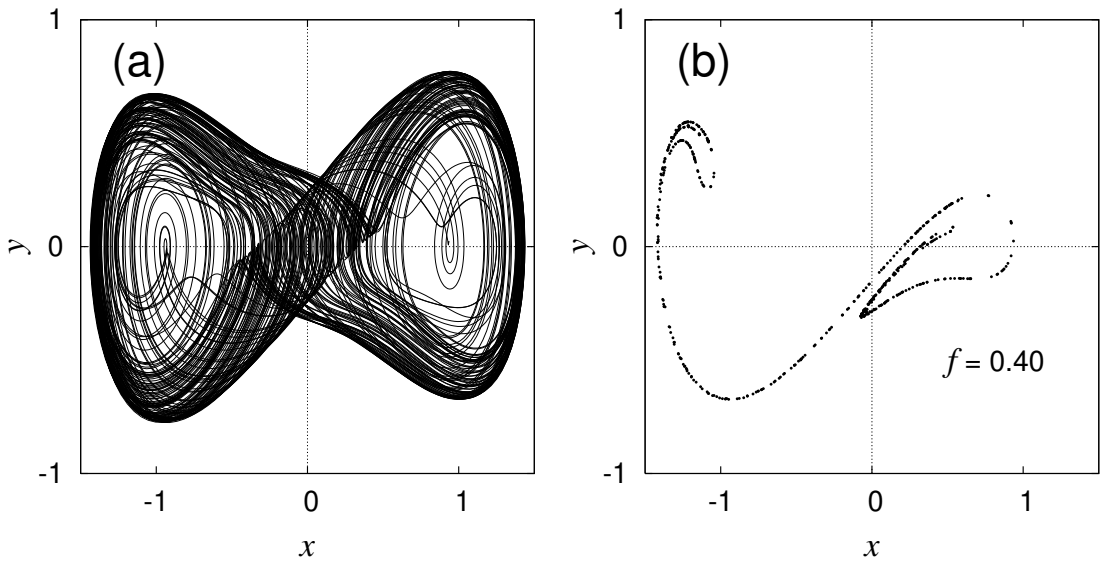
Duffing oscillator

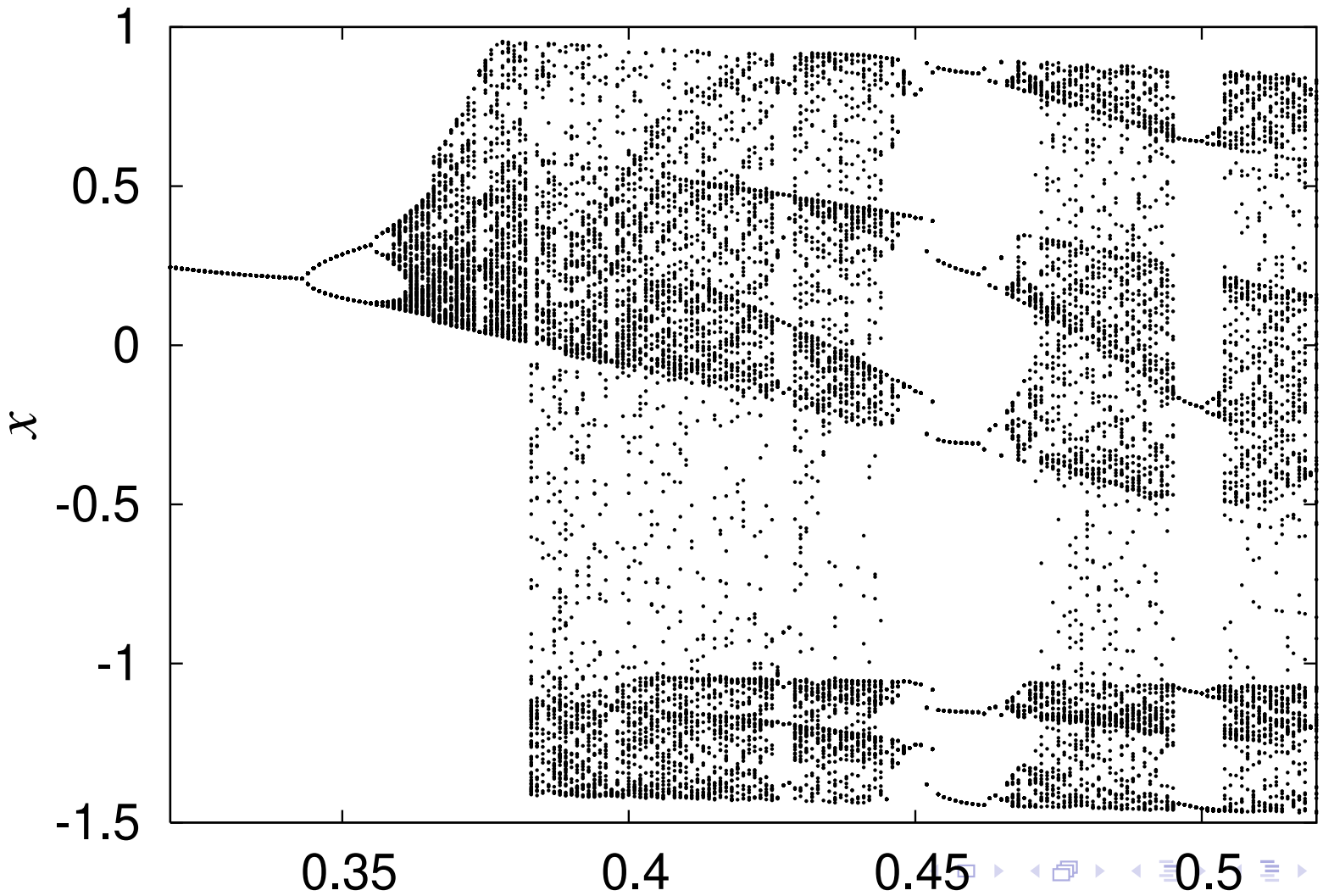
Duffing oscillator

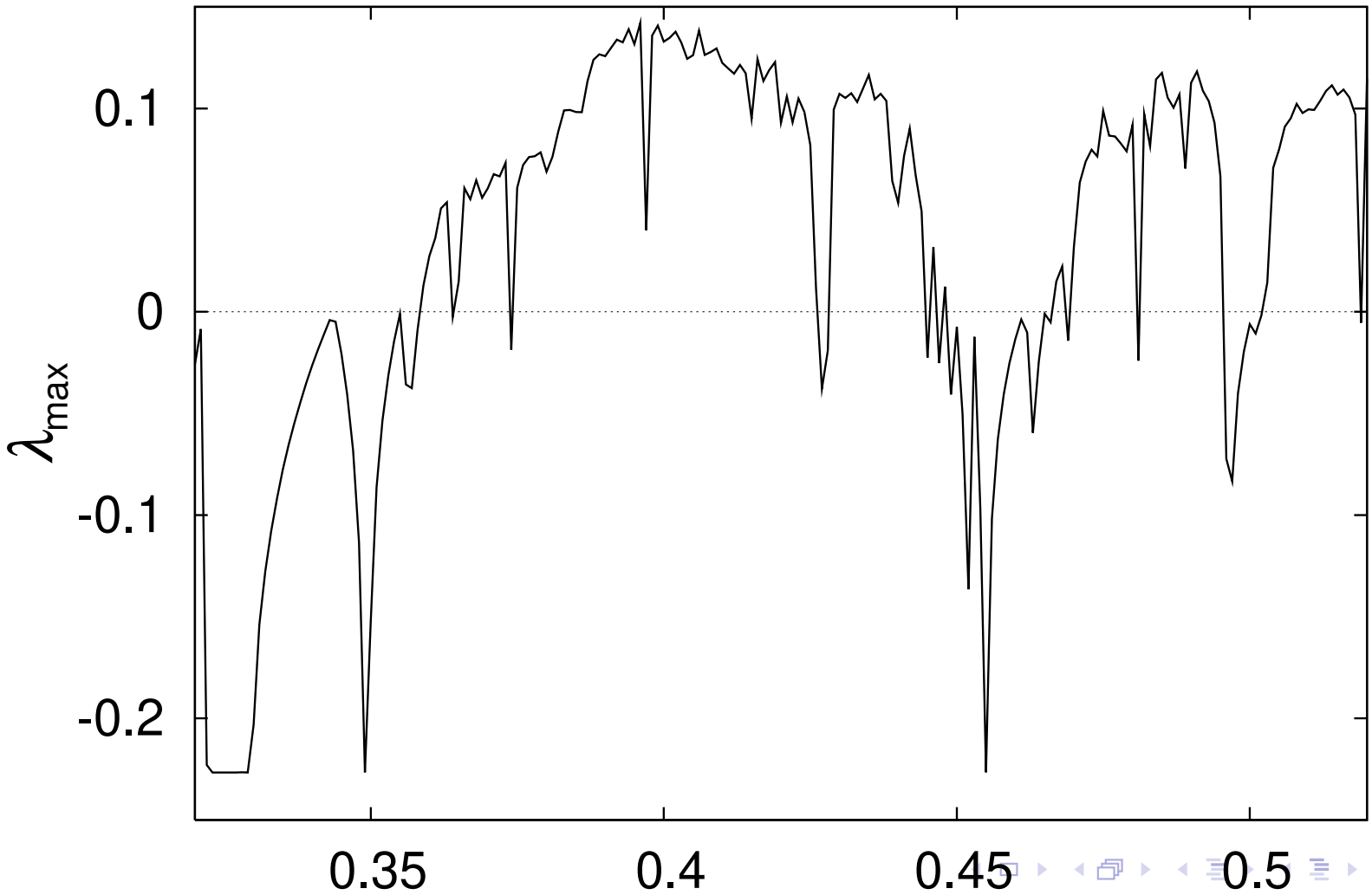
$$\ddot{x} + \alpha\dot{x} + \omega_0^2 x + \beta x^3 = f \sin \omega t$$

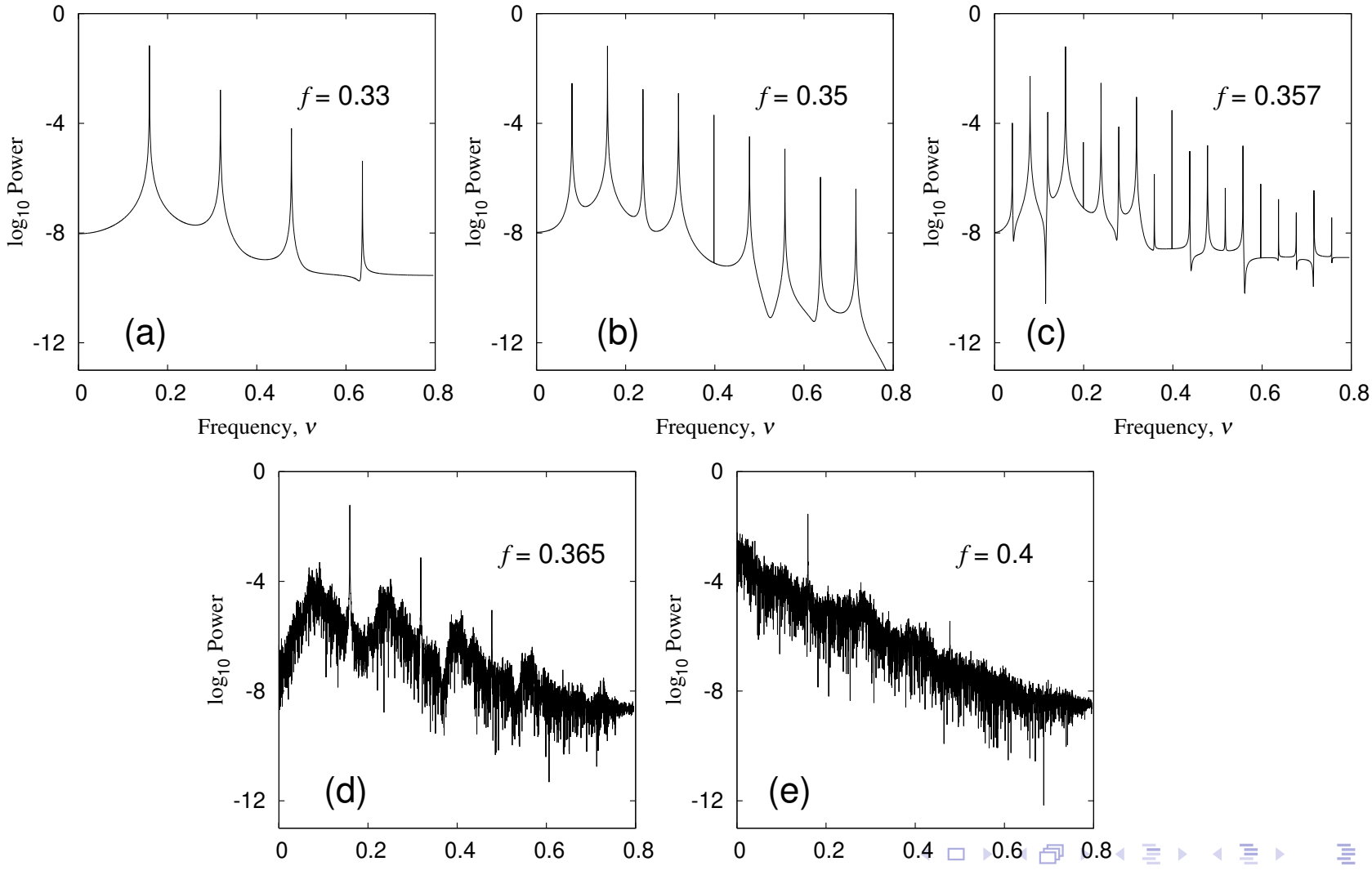
$$\alpha = 0.5, \quad \omega_0^2 = -1, \quad \omega = 1$$


(a)
$f = 0$
(b)
$f = 0.33$
(c)
$f = 0.35$
(d)
$f = 0.357$
(e)
$f = 0.365$
(f)
$f = 0.365$
x
y
1
0
-1


(a)
(b)
1
0
-1
y
x
f = 0.40


1
0.5
0
-0.5
-1
-1.5
x
0.35
0.4
0.45
0.5


0.1
0
-0.1
-0.2
λ_{max}
0.35
0.4
0.45
0.5


0
-4
-8
-12
$\log_{10}$ Power
$f = 0.33$
(a)
0 0.2 0.4 0.6 0.8
Frequency, ν
$f = 0.35$
(b)
$f = 0.357$
(c)
$f = 0.365$
(d)
$f = 0.4$
(e)