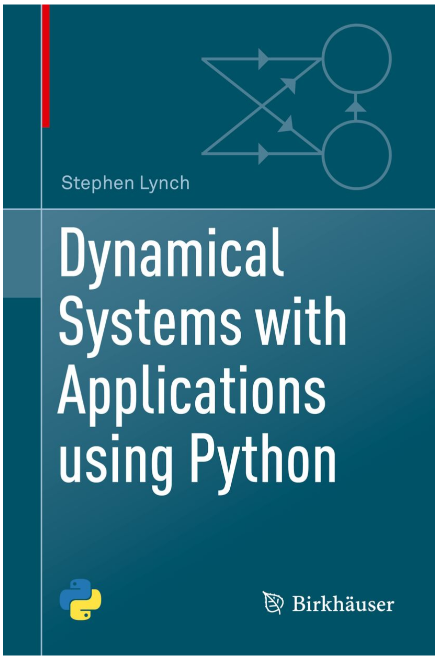
Birkhäuser

birkhauser-science.de



1st ed. 2018, XVI, 665 p. 277 illus., 118 illus. in color.

Printed book

Hardcover

79,99 € | £69.99 | $89.99
[1]85,59 € (D) | 87,99 € (A) | CHF 94,50

eBook

67,82 € | £55.99 | $69.99
[2]67,82 € (D) | 67,82 € (A) | CHF 75,50

Available from your library or springer.com/shop

MyCopy [3]

Printed eBook for just
€ | $ 24.99
springer.com/mycopy

Stephen Lynch

Dynamical Systems with Applications using Python

- **Designed for a broad audience of students in applied mathematics, physics, and engineering**
- **Represents dynamical systems with popular Python libraries like sympy, numpy, and matplotlib**
- **Explores a variety of advanced topics in dynamical systems, like neural networks, fractals, and nonlinear optics, at an undergraduate level**

This textbook provides a broad introduction to continuous and discrete dynamical systems. With its hands-on approach, the text leads the reader from basic theory to recently published research material in nonlinear ordinary differential equations, nonlinear optics, multifractals, neural networks, and binary oscillator computing. Dynamical Systems with Applications Using Python takes advantage of Python's extensive visualization, simulation, and algorithmic tools to study those topics in nonlinear dynamical systems through numerical algorithms and generated diagrams. After a tutorial introduction to Python, the first part of the book deals with continuous systems using differential equations, including both ordinary and delay differential equations. The second part of the book deals with discrete dynamical systems and progresses to the study of both continuous and discrete systems in contexts like chaos control and synchronization, neural networks, and binary oscillator computing. These later sections are useful reference material for undergraduate student projects. The book is rounded off with example coursework to challenge students' programming abilities and Python-based exam questions. This book will appeal to advanced undergraduate and graduate students, applied mathematicians, engineers, and researchers in a range of disciplines, such as biology, chemistry, computing, economics, and physics.

Birkhäuser

Stephen Lynch

Dynamical Systems with Applications using Python

Birkhäuser

Stephen Lynch
Manchester Metropolitan University
Manchester, UK

ISBN 978-3-319-78144-0 ISBN 978-3-319-78145-7 (eBook)
https://doi.org/10.1007/978-3-319-78145-7

Library of Congress Control Number: 2018952351

Mathematics Subject Classification (2010): 00A05, 00A69, 37-01, 34-01, 34Cxx, 34Dxx, 34H10, 34K18, 37-04, 37Nxx, 68U10, 78A60, 92B20

This book is published under the imprint Birkhäuser, www.birkhauser-science.com by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This book provides an introduction to the theory of dynamical systems with the aid of Python. It is written for both senior undergraduates and graduate students. Chapter 1 provides a tutorial introduction to Python—new users should go through this chapter carefully while those moderately familiar and experienced users will find this chapter a useful source of reference. The first part of the book deals with continuous systems using differential equations, including both ordinary and delay differential equations (Chapters 2–12), the second part is devoted to the study of discrete systems (Chapters 13–17), and Chapters 18–21 deal with both continuous and discrete systems. Chapter 22 gives examples of coursework and also lists three Python-based examinations to be sat in a computer laboratory with access to Python. Chapter 23 lists answers to all of the exercises given in the book. It should be pointed out that dynamical systems theory is not limited to these topics but also encompasses partial differential equations, integral and integro-differential equations, and stochastic systems, for instance. References [1–6] given at the end of the Preface provide more information for the interested reader. The author has gone for breadth of coverage rather than fine detail and theorems with proofs are kept at a minimum. The material is not clouded by functional analytic and group theoretical definitions, and so is intelligible to readers with a general mathematical background. Some of the topics covered are scarcely covered elsewhere. Most of the material in Chapters 9–12 and 16–21 is at postgraduate level and has been influenced by the author's own research interests. There is more theory in these chapters than in the rest of the book since it is not easily accessed anywhere else. It has been found that these chapters are especially useful as reference material for senior undergraduate project work. The theory in other chapters of the book is dealt with more comprehensively in other texts, some of which may be found in the references section of the corresponding chapter. The book has a very hands-on approach and takes the reader from the basic theory right through to recently published research material.

Python is extremely popular with a wide range of researchers from all sorts of disciplines; it has a very user-friendly interface and has extensive visualization and numerical computation capabilities. It is an ideal package to adopt for the study of nonlinear dynamical systems; the numerical algorithms work very quickly, and complex pictures can be plotted within seconds.

The first chapter provides an efficient tutorial introduction to Python. Simple Python programming is introduced using three basic programming structures: defining functions, for loops, and if, then, else constructs. New users will find the tutorials will enable them to become familiar with Python within a few days. Both engineering and mathematics students appreciate this method of teaching and I have found that it generally works well with one staff member to about twenty students in a computer laboratory. In most cases, I have chosen to list the Python commands at the end of each chapter; this avoids unnecessary cluttering in the text. The Python programs have been kept as simple as possible and should run under later versions of the package. All Python files for the book (including updates and extra files) can even be downloaded from the Web via GitHub at:

https://github.com/springer-math/dynamical-systems-with-applications-using-python

Readers will find that they can reproduce the figures given in the text, and then it is not too difficult to change parameters or equations to investigate other systems.

Chapters 2–12 deal with continuous dynamical systems. Chapters 2 and 3 cover some theory of ordinary differential equations and applications to models in the real world are given. The theory of differential equations applied to chemical kinetics and electric circuits is introduced in some detail. The memristor is introduced and one of the most remarkable stories in the history of mathematics is relayed. Chapter 2 ends with the existence and uniqueness theorem for the solutions of certain types of differential equations. The theory behind the construction of phase plane portraits for two-dimensional systems is dealt with in Chapter 3. Applications are taken from chemical kinetics, economics, electronics, epidemiology, mechanics, and population dynamics. The modeling of the populations of interacting species is discussed in some detail in Chapter 4 and domains of stability are discussed for the first time. Limit cycles, or isolated periodic solutions, are introduced in Chapter 5. Since we live in a periodic world, these are the most common type of solution found when modeling nonlinear dynamical systems. They appear extensively when modeling both the technological and natural sciences. Hamiltonian, or conservative, systems and stability are discussed in Chapter 6, and Chapter 7 is concerned with how planar systems vary depending upon a parameter. Bifurcation, bistability, multistability, and normal forms are discussed.

The reader is first introduced to the concept of chaos in continuous systems in Chapters 8 and 9, where three-dimensional systems and Poincaré maps are investigated. These higher-dimensional systems can exhibit strange attractors and chaotic dynamics. One can rotate the three-dimensional objects in Python and plot time series plots to get a better understanding of the dynamics involved. Once again, the theory can be applied to chemical kinetics (including stiff systems), electric circuits, and epidemiology; a simplified model for the weather is also briefly discussed. Chapter 9 deals with Poincaré first return maps that can be used to untangle complicated interlacing trajectories in higher-dimensional spaces. A periodically driven nonlinear pendulum is also investigated by means of a nonautonomous differential equation. Both local and global bifurcations are investigated in Chapter 10. The main results and statement of the famous second part of David Hilbert's sixteenth problem are listed in Chapter 11. In order to understand these results, Poincaré compactification is introduced. The study of continuous systems ends with one of the authors specialities—limit cycles of Liénard systems. There is some detail on Liénard systems, in particular, in this part of the book, but they do have a ubiquity for systems in the plane. Chapter 12 provides an introduction to delay differential equations with applications in biology and nonlinear optics.

Chapters 13–17 deal with discrete dynamical systems. Chapter 13 starts with a general introduction to iteration and linear recurrence (or difference) equations. The bulk of the chapter is concerned with the Leslie model used to investigate the population of a single species split into different age classes. Harvesting and culling policies are then investigated and optimal solutions are sought. Nonlinear discrete dynamical systems are dealt with in Chapter 14. Bifurcation diagrams, chaos, intermittency, Lyapunov exponents, periodicity, quasiperiodicity, and universality are some of the topics introduced. The theory is then applied to real-world problems from a broad range of disciplines including population dynamics, biology, economics, nonlinear optics, and neural networks. Chapter 15 is concerned with complex iterative maps in the Argand plane, where Julia sets and the now-famous Mandelbrot set are plotted. Basins of attraction are investigated for these complex systems and Newton fractals are introduced. As a simple introduction to optics, electromagnetic waves and Maxwell's equations are studied at the beginning of Chapter 16. Complex iterative equations are used to model the propagation of light waves through nonlinear optical fibers. A brief history of nonlinear bistable optical resonators is discussed and the simple fiber ring resonator is analyzed in particular. Chapter 16 is devoted to the study of these optical resonators, and there is discussion on phenomena such as bistability, chaotic attractors, feedback, hysteresis, instability, linear stability analysis, multistability, nonlinearity, and steady states. The first and second iterative methods are defined in this chapter. Some simple fractals may be constructed using pencil and paper in Chapter 17, and the concept of fractal dimension is introduced. Fractals may be thought of as identical motifs repeated on ever-reduced scales. Unfortunately, most of the fractals appearing in nature are not homogeneous but are more heterogeneous, hence the need for the multifractal theory given later in the chapter. It has been found that the distribution of stars and galaxies in our universe is multifractal, and there is even evidence of multifractals in rainfall, stock markets, and heartbeat rhythms. Applications in geoscience, materials science, microbiology, and image processing are briefly discussed. Chapter 18 provides a brief introduction to image processing which is being used more and more by a diverse range of scientific disciplines, especially medical imaging. The fast Fourier transform is introduced and has a wide range of applications throughout the realms of science.

Chapter 19 is devoted to the new and exciting theory behind chaos control and synchronization. For most systems, the maxim used by engineers in the past has been "stability good, chaos bad," but more and more nowadays this is being replaced with "stability good, chaos better." There are exciting and novel applications in cardiology, communications, engineering, laser technology, and space research, for example. A brief introduction to the enticing field of neural networks is presented in Chapter 20. Imagine trying to make a computer mimic the human brain. One could ask the question: In the future will it be possible for computers to think and even be conscious? The human brain will always be more powerful than traditional, sequential, logic-based digital computers and scientists are trying to incorporate some features of the brain into modern computing. Neural networks perform through learning and no underlying equations are required. Mathematicians and computer scientists are attempting to mimic the way neurons work together via synapses; indeed, a neural network can be thought of as a crude multidimensional model of the human brain. The expectations are high for future applications in a broad range of disciplines. Neural networks are already being used in machine learning and pattern recognition (computer vision, credit card fraud, prediction and forecasting, disease recognition, facial and speech recognition), the consumer home entertainment market, psychological profiling, predicting wave over-topping events, and control problems, for example. They also provide a parallel architecture allowing for very fast computational and response times. In recent years, the disciplines of neural networks and nonlinear dynamics have increasingly coalesced and a new branch of science called neurodynamics is emerging. Lyapunov functions can be used to determine the stability of certain types of neural network. There is also evidence of chaos, feedback, nonlinearity, periodicity, and chaos synchronization in the brain.

Chapter 21 focuses on binary oscillator computing, the subject of UK, International, and Taiwanese patents. The author and his co-inventor, Jon

Borresen, came up with the idea when modeling connected biological neurons. Binary oscillator technology can be applied to the design of arithmetic logic units (ALUs), memory, and other basic computing components. It has the potential to provide revolutionary computational speed-up, energy saving, and novel applications and may be applicable to a variety of technological paradigms including biological neurons, complementary metal-oxide-semiconductor (CMOS), memristors, optical oscillators, and superconducting materials. The research has the potential for MMU and industrial partners to develop super fast, low-power computers and may provide an assay for neuronal degradation for brain malfunctions such as Alzheimer's, epilepsy, and Parkinson's disease!

Examples of coursework and three examination-type papers are listed in Chapter 22, and a complete set of solutions for the book is listed in Chapter 23.

Both textbooks and research papers are presented in the list of references. The textbooks can be used to gain more background material, and the research papers have been given to encourage further reading and independent study.

This book is informed by the research interests of the author, which are currently nonlinear ordinary differential equations, nonlinear optics, multifractals, neural networks, and binary oscillator computing. Some references include recently published research articles by the author along with two patents.

The prerequisites for studying dynamical systems using this book are undergraduate courses in linear algebra, real and complex analysis, calculus, and ordinary differential equations; a knowledge of a computer language such as Basic, C, or Fortran would be beneficial but not essential.

Recommended Textbooks

[1] H.P Langtangen and A. Logg, Solving PDEs in Python: The FEniCS Tutorial I (Simula SpringerBriefs on Computing), Springer, New York, 2017.

[2] B. Bhattacharya and M. Majumdar, *Random Dynamical Systems in Finance*, Chapman & Hall/CRC, New York, 2016.

[3] L.C. de Barros, R.C. Bassanezi and W.A. Lodwick, *A First Course in Fuzzy Logic, Fuzzy Dynamical Systems, and Biomathematics: Theory and Applications*, Springer, New York, 2016.

[4] V. Volterra, *Theory of Functionals and of Integral and Integro-Differential Equations*, Dover Publications, New York, 2005.

[5] J. Mallet-Paret (Editor), J. Wu (Editor), H. Zhu (Editor), Y. Yi (Editor), *Infinite Dimensional Dynamical Systems (Fields Institute Communications)*, Springer, New York, 2013.

[6] C. Bernido, M.V. Carpio-Bernido, M. Grothaus et al., *Stochastic and Infinite Dimensional Analysis*, Birkhäuser, New York, 2016.

Special thanks go to Ben Nuttall (Python guru), Community Manager, the Raspberry Pi Foundation, Cambridge, UK (www.raspberrypi.org), for reviewing this book. I would also like to express my sincere thanks to all of the reviewers of this book and the other editions of my books. As always, thanks also go to Birkhäuser and Springer, especially Samuel DiBella (Assistant Editor, Springer Nature). Finally, thanks to my family and especially my wife Gaynor, and our children, Sebastian and Thalia, for their continuing love, inspiration, and support.

Manchester, UK Stephen Lynch FIMA SFHEA

Contents

Appendix A

Index of Python Programs

Readers can download the Python program files via GitHub:

https://github.com/springer-math/dynamical-systems-with-applications-using-python

These files will be kept up-to-date and extra files will be added in the forthcoming years.

A.1 IDLE Python Programs

These files include solutions to the Exercises listed in Chapter 1.

```
euclid_algorithm.py --- See Exercise 10.
F2C.py --- See Exercise 1(a).
F2K.py --- Converts degrees Fahrenheit to Kelvin.
fibonacci.py --- Lists first n terms of the Fibonacci sequence.
fmu.py --- The logistic function.
fractal_tree.py --- Plots a fractal tree.
fractal_tree_color.py --- Plots a color fractal tree.
grade.py --- Converts a score to a grade.
guess_number.py --- Guess the number game.
koch_snowflake.py --- See Exercise 1(d).
koch_square.py --- Plots a Koch square fractal.
Pythag_Triples.py --- See Exercise 1(c).
sierpinski.py --- Plots a Sierpinski triangle fractal.
sierpinski_square.py --- Plots a Sierpinski square fractal.
```


S. Lynch, *Dynamical Systems with Applications using Python*,
https://doi.org/10.1007/978-3-319-78145-7

```
sum_primes.py --- See Exercise 1(b).
sum_n.py --- Sums the natural numbers to n.
```

A.2 Anaconda Python Programs

If you have difficulty with the animation programs in Spyder, you have to change the backend to run an animation in the IPython console. You can do that by running

```
In[1]: %matplotlib qt5
```

before the animation. If you don't want to use this command every time, you can go to: Tools, Preferences, IPython Console, Graphics, Backend, and change it from "Inline" to "Automatic."

```
Program_01a.py --- Solve a simple ODE.
Program_01b.py --- Solve a second order ODE.
Program_01c.py --- Plot two curves on one graph.
Program_01d.py --- Subplots.
Program_01e.py --- Surface and contour plot in 3D.
Program_01f.py --- A parametric curve in 3D.
Program_01g.py --- Animation of a simple curve.

Program_02a.py --- Solve a separable ODE.
Program_02b.py --- Solve the logistic ODE.
Program_02c.py --- Power series solution.
Program_02d.py --- Power series solution for van der Pol.
Program_02e.py --- Plot series solution against numerical
                   solution.
Program_02f.py --- Solve a linear first order ODE.
Program_02g.py --- Solve a linear second order ODE.

Program_03a.py --- Plot the phase portrait of a linear system.
Program_03b.py --- Plot the phase portrait of a nonlinear
                   system.
Program_03c.py --- Finding critical points.

Program_04a.py --- Phase portrait and time series of Holling-
                   Tanner model.
```

```
Program_05a.py --- Limit cycle of a Fitzhugh-Nagumo system.
Program_05b.py --- Approximate and numerical solutions to ODEs.
Program_05c.py --- Error between one-term and numerical
                   solution.
Program_05d.py --- Lindstedt-Poincare technique.

Program_06a.py --- Contour plot.
Program_06b.py --- Surface plot.

Program_07a.py --- Animation of a simple curve.
Program_07b.py --- Animation of a subcritical Hopf bifurcation.
Program_07c.py --- Animation of a SNIC bifurcation.

Program_08a.py --- The Rossler attractor.
Program_08b.py --- The Lorenz Attractor.
Program_08c.py --- The Belousov-Zhabotinsky reaction.
Program_08d.py --- Animation of a Chua circuit bifurcation.

Program_09a.py --- Simple Poincare return map.
Program_09b.py --- Hamiltonian with two degrees of freedom plot.
Program_09c.py --- Phase portrait and Poincare map for the
                   Duffing system.
Program_09d.py --- Bifurcation diagram of Duffing equation.

Program_10a.py --- Computing Lyapunov quantities.
Program_10b.py --- Division algorithm for multivariate
                   polynomials.
Program_10c.py --- S-polynomial.
Program_10d.py --- Computing the Groebner basis.
Program_10e.py --- Computing Groebner basis of Lyapunov
                   quantities.
Program_10f.py --- Animation of a homoclinic limit cycle
                   bifurcation.
Program_10g.py --- Animation of a homoclinic limit cycle
                   bifurcation.

Program_11a.py --- Animation of a Lienard limit cycle.

Program_12a.py --- The method of steps.
Program_12b.py --- Plot of solution by method of steps.
Program_12c.py --- The Mackey-Glass DDE.
Program_12d.py --- The Lang-Kobayashi DDEs.
```

```
Program_13a.py --- Computing bank interest.
Program_13b.py --- Solving a second order recurrence relation.
Program_13c.py --- The Leslie matrix, eigenvalues and
                   eigenvectors.

Program_14a.py --- Graphical iteration of the tent map.
Program_14b.py --- Bifurcation diagram of the logistic map.
Program_14c.py --- Computing Lyapunov exponents for the
                   logistic map.
Program_14d.py --- Iteration of the Henon map.
Program_14e.py --- Lyapunov exponents of the Henon map.

Program_15a.py --- Point plot for a Julia set.
Program_15b.py --- Colormap of a Julia set.
Program_15c.py --- Color Mandelbrot set.
Program_15d.py --- Color Newton fractal Julia set.

Program_16a.py --- Intersection of implicit curves.
Program_16b.py --- Chaotic Attractor of the Ikeda map
Program_16c.py --- Bifurcation diagram of the Ikeda map.

Program_17a.py --- The Koch curve.
Program_17b.py --- Chaos game and the Sierpnski triangle.
Program_17c.py --- Barnsley's fern.
Program_17d.py --- Subplots of tau, D_q and f(alpha)
                   multifractal spectra.

Program_18a.py --- Generating a multifractal image
Program_18b.py --- Counting pixels in a color image.
Program_18c.py --- Image and statistical analysis on
                   microbes.png
Program_18d.py --- Fast Fourier transform of a noisy signal
Program_18e.py --- Iterative map and power spectra
Program_18f.py --- Fast Fourier transform of Lena image
Program_18g.py --- Edge detection in Lena image

Program_19a.py --- Chaos control in the logistic map.
Program_19b.py --- Chaos control in the Henon map.
Program_19c.py --- Chaos synchronization between two Lorenz
                   systems.
Program_19d.py --- Generalized synchronization.

Program_20a.py --- The generalized delta learning rule.
```

```
Program_20b.py --- The discrete Hopfield network.
Program_20c.py --- Iteration of a minimal chaotic neuromodule.
Program_20d.py --- Bifurcation diagram of neuromodule.

Program_21a.py --- The Hodgkin-Huxley equations.
Program_21b.py --- The Fitzhugh-Nagumo half-adder.
Program_21c.py --- Phase portrait Josephson junction limit
                   cycle.
Program_21d.py --- Animated Josephson junction limit cycle.
Program_21e.py --- Pinched hysteresis of a memristor.
```

Index

S. Lynch, *Dynamical Systems with Applications using Python*,
https://doi.org/10.1007/978-3-319-78145-7