MAA Notes #76

MAA
MATHEMATICAL ASSOCIATION OF AMERICA

THE BEAUTY OF FRACTALS:

Six Different Views

Denny Gulick and Jon Scott, Editors

The Beauty of Fractals

Six Different Views

Library of Congress Control Number 2010936090

Print edition ISBN 978-0-88385-186-9

Electronic edition ISBN 978-0-88385-971-1

Printed in the United States of America

Current Printing (last digit):
10 9 8 7 6 5 4 3 2 1

The Beauty of Fractals

Six Different Views

Edited by

Denny Gulick
University of Maryland

and

Jon Scott
Montgomery College

Published and Distributed by
The Mathematical Association of America

The MAA Notes Series, started in 1982, addresses a broad range of topics and themes of interest to all who are involved with undergraduate mathematics. The volumes in this series are readable, informative, and useful, and help the mathematical community keep up with developments of importance to mathematics.

MAA Notes

14. Mathematical Writing, by *Donald E. Knuth, Tracy Larrabee, and Paul M. Roberts.*
16. Using Writing to Teach Mathematics, *Andrew Sterrett,* Editor.
17. Priming the Calculus Pump: Innovations and Resources, Committee on Calculus Reform and the First Two Years, a subcomittee of the Committee on the Undergraduate Program in Mathematics, *Thomas W. Tucker,* Editor.
18. Models for Undergraduate Research in Mathematics, *Lester Senechal,* Editor.
19. Visualization in Teaching and Learning Mathematics, Committee on Computers in Mathematics Education, *Steve Cunningham and Walter S. Zimmermann,* Editors.
20. The Laboratory Approach to Teaching Calculus, *L. Carl Leinbach et al.,* Editors.
21. Perspectives on Contemporary Statistics, *David C. Hoaglin and David S. Moore,* Editors.
22. Heeding the Call for Change: Suggestions for Curricular Action, *Lynn A. Steen,* Editor.
24. Symbolic Computation in Undergraduate Mathematics Education, *Zaven A. Karian,* Editor.
25. The Concept of Function: Aspects of Epistemology and Pedagogy, *Guershon Harel and Ed Dubinsky,* Editors.
26. Statistics for the Twenty-First Century, *Florence and Sheldon Gordon,* Editors.
27. Resources for Calculus Collection, Volume 1: Learning by Discovery: A Lab Manual for Calculus, *Anita E. Solow,* Editor.
28. Resources for Calculus Collection, Volume 2: Calculus Problems for a New Century, *Robert Fraga,* Editor.
29. Resources for Calculus Collection, Volume 3: Applications of Calculus, *Philip Straffin,* Editor.
30. Resources for Calculus Collection, Volume 4: Problems for Student Investigation, *Michael B. Jackson and John R. Ramsay,* Editors.
31. Resources for Calculus Collection, Volume 5: Readings for Calculus, *Underwood Dudley,* Editor.
32. Essays in Humanistic Mathematics, *Alvin White,* Editor.
33. Research Issues in Undergraduate Mathematics Learning: Preliminary Analyses and Results, *James J. Kaput and Ed Dubinsky,* Editors.
34. In Eves Circles, *Joby Milo Anthony,* Editor.
35. Youre the Professor, What Next? Ideas and Resources for Preparing College Teachers, The Committee on Preparation for College Teaching, *Bettye Anne Case,* Editor.
36. Preparing for a New Calculus: Conference Proceedings, *Anita E. Solow,* Editor.
37. A Practical Guide to Cooperative Learning in Collegiate Mathematics, *Nancy L. Hagelgans, Barbara E. Reynolds, SDS, Keith Schwingendorf, Draga Vidakovic, Ed Dubinsky, Mazen Shahin, G. Joseph Wimbish, Jr.*
38. Models That Work: Case Studies in Effective Undergraduate Mathematics Programs, *Alan C. Tucker,* Editor.
39. Calculus: The Dynamics of Change, CUPM Subcommittee on Calculus Reform and the First Two Years, *A. Wayne Roberts,* Editor.
40. Vita Mathematica: Historical Research and Integration with Teaching, *Ronald Calinger,* Editor.
41. Geometry Turned On: Dynamic Software in Learning, Teaching, and Research, *James R. King and Doris Schattschneider,* Editors.
42. Resources for Teaching Linear Algebra, *David Carlson, Charles R. Johnson, David C. Lay, A. Duane Porter, Ann E. Watkins, William Watkins,* Editors.
43. Student Assessment in Calculus: A Report of the NSF Working Group on Assessment in Calculus, *Alan Schoenfeld,* Editor.
44. Readings in Cooperative Learning for Undergraduate Mathematics, *Ed Dubinsky, David Mathews, and Barbara E. Reynolds,* Editors.
45. Confronting the Core Curriculum: Considering Change in the Undergraduate Mathematics Major, *John A. Dossey,* Editor.
46. Women in Mathematics: Scaling the Heights, *Deborah Nolan,* Editor.
47. Exemplary Programs in Introductory College Mathematics: Innovative Programs Using Technology, *Susan Lenker,* Editor.
48. Writing in the Teaching and Learning of Mathematics, *John Meier and Thomas Rishel.*
49. Assessment Practices in Undergraduate Mathematics, *Bonnie Gold,* Editor.
50. Revolutions in Differential Equations: Exploring ODEs with Modern Technology, *Michael J. Kallaher,* Editor.

51. Using History to Teach Mathematics: An International Perspective, *Victor J. Katz*, Editor.
52. Teaching Statistics: Resources for Undergraduate Instructors, *Thomas L. Moore*, Editor.
53. Geometry at Work: Papers in Applied Geometry, *Catherine A. Gorini*, Editor.
54. Teaching First: A Guide for New Mathematicians, *Thomas W. Rishel*.
55. Cooperative Learning in Undergraduate Mathematics: Issues That Matter and Strategies That Work, *Elizabeth C. Rogers, Barbara E. Reynolds, Neil A. Davidson, and Anthony D. Thomas*, Editors.
56. Changing Calculus: A Report on Evaluation Efforts and National Impact from 1988 to 1998, *Susan L. Ganter*.
57. Learning to Teach and Teaching to Learn Mathematics: Resources for Professional Development, *Matthew Delong and Dale Winter.*
58. Fractals, Graphics, and Mathematics Education, Benoit Mandelbrot and Michael Frame, Editors.
59. Linear Algebra Gems: Assets for Undergraduate Mathematics, *David Carlson, Charles R. Johnson, David C. Lay, and A. Duane Porter*, Editors.
60. Innovations in Teaching Abstract Algebra, *Allen C. Hibbard and Ellen J. Maycock*, Editors.
61. Changing Core Mathematics, *Chris Arney and Donald Small*, Editors.
62. Achieving Quantitative Literacy: An Urgent Challenge for Higher Education, *Lynn Arthur Steen.*
64. Leading the Mathematical Sciences Department: A Resource for Chairs, *Tina H. Straley, Marcia P. Sward, and Jon W. Scott*, Editors.
65. Innovations in Teaching Statistics, *Joan B. Garfield*, Editor.
66. Mathematics in Service to the Community: Concepts and models for service-learning in the mathematical sciences, *Charles R. Hadlock*, Editor.
67. Innovative Approaches to Undergraduate Mathematics Courses Beyond Calculus, *Richard J. Maher*, Editor.
68. From Calculus to Computers: Using the last 200 years of mathematics history in the classroom, Amy Shell-Gellasch and Dick Jardine, Editors.
69. A Fresh Start for Collegiate Mathematics: Rethinking the Courses below Calculus, *Nancy Baxter Hastings*, Editor.
70. Current Practices in Quantitative Literacy, *Rick Gillman*, Editor.
71. War Stories from Applied Math: Undergraduate Consultancy Projects, *Robert Fraga*, Editor.
72. Hands On History: A Resource for Teaching Mathematics, *Amy Shell-Gellasch*, Editor.
73. Making the Connection: Research and Teaching in Undergraduate Mathematics Education, *Marilyn P. Carlson and Chris Rasmussen*, Editors.
74. Resources for Teaching Discrete Mathematics: Classroom Projects, History Modules, and Articles, *Brian Hopkins*, Editor.
75. The Moore Method: A Pathway to Learner-Centered Instruction, *Charles A. Coppin, W. Ted Mahavier, E. Lee May, and G. Edgar Parker*.
76. The Beauty of Fractals: Six Different Views, *Denny Gulick and Jon Scott*, Editors.

MAA Service Center
P.O. Box 91112
Washington, DC 20090-1112
1-800-331-1MAA FAX: 1-301-206-9789

Preface

Fractals came onto the stage in the 1970's with the emergence of the Mandelbrot set, with its incredibly complicated and interesting boundary. During the 1980's a number of books appeared, including most especially by Mandelbrot, Barnsley and Devaney, that gave a mathematical background for fractals that made fractals accessible to both students and teachers. More recently, as computers and their users have become more sophisticated, the domain of fractals has broadened, from art to scientific application to mathematical analysis. In particular, students in high school as well as college are often introduced to fractals and fractal concepts. The present volume includes six essays related to fractals, with perspectives different enough to give you a taste of the breadth of the subject.

Each essay is self-contained and expository. Moreover, each of the essays is intended to be accessible to a broad audience that includes college teachers, high school teachers, advanced undergraduate students, and others who wish to learn or teach about topics in fractals that are not regularly in textbooks on fractals.

Next is a brief overview of each essay; together these overviews should give you quite different views of the topic of fractals.

The volume begins with "Mathscapes—Fractal Geometry," by Anne M. Burns. Burns, who is an artist as well as a mathematician, discusses several ways of modeling on the computer such fractal objects as plant growth and trees, clouds and mountains. The algorithms that Burns uses to create such fascinating and beautiful fractal scenery include stochastic matrices, simple recursion, and a probabilistic method, all of which are accessible to students in a variety of courses from mathematics to programming to graphics.

The second essay is "Chaos, Fractals, and Tom Stoppard's *Arcadia*," by Robert Devaney. We don't often find mathematical themes in works meant for the theater, but with this essay, Devaney introduces us to a notable exception. Indeed, ideas from fractal geometry and chaos theory are "center stage" in *Arcadia*. As Devaney summarizes the main action of the play, he reminds us of the basic mathematics of iterated function systems (the chaos game) and the dynamics of the logistic function, both of which are very familiar in the study of fractals. In addition, he suggests that by means of the play, teachers can provide a rich interdisciplinary experience for their students.

The third essay is "Excursions Through a Forest of Golden Fractal Trees," by T. D. Taylor. In this article, Taylor explores surprising and beautiful connections between fractal geometry and the geometry associated with the golden ratio. In particular, Taylor discusses four self-contacting symmetric binary fractal trees that scale with the golden ratio. The article includes new variations on such familiar fractals as the Cantor set, the Koch curve and Koch snowflake. And, as Taylor points out, throughout the paper there are "wonderful exercises involving trigonometry, geometric series, the scaling nature of fractal trees, and the many special equations involving the golden ratio."

The fourth essay is "Exploring Fractal Dimension, Area, and Volume," by Mary Ann Connors. In this essay, Connors discusses basic properties of some of the most famous and picturesque fractals. They include the Sierpiński Triangle (or Gasket) and the 3-dimensional analogue of the Sierpiński Triangle, the Harter-Heighway Dragon, Sierpiński's Carpet, the Koch Snowflake and a 3-dimensional analogue of the Sierpiński Carpet, and finally the Mandelbrot Set.

The fifth essay is "Points in Sierpiński-like Fractals," by Sandra Fillebrown, Joseph Pizzica, Vincent Russo, and Scott Fillebrown. The impetus for the essay is the Sierpiński Triangle, one of the most famous of the classical fractals appearing a century ago, which can be defined by three special contractions of the plane. The present essay defines and studies "Sierpiński-like" fractals, which are sets in the plane that also can be defined by three special contractions, however utilizing reflections and rotations of those contractions. The essay shows how to generate rules based on the binary representations in order to determine whether a point is or is not in a given Sierpiński-like fractal.

The final essay is “Fractals in the 3-Body Problem Via Symplectic Integration,” by Daniel Hemberger and James A. Walsh. The 3-body problem, which seeks the positions and velocities of three bodies over time, was studied by Henri Poincaré in the late 19th century, and gave rise to what became the subject of dynamical systems. In this essay, Hemberger and Walsh investigate the 3-body problem through the notion of symplectic maps (defined in the essay). With an elementary approach they discuss why symplectic maps can be very effective for numerical integration of Hamiltonian systems of differential equations. Then they use the information gained to investigate fractals that arise in the study of the 3-body problem.

The editors of this volume would like to thank the MAA for including a contributed paper session on fractals and chaos at the annual meeting in January 2007, from which the six essays included here are derived. The editors would also like to thank the Series editor, Stephen Maurer, and the reviewers, all of whom made many helpful suggestions. Finally, the editors acknowledge with great appreciation the careful work by the MAA production staff, most especially Beverly Ruedi, Elaine Pedreira, and Carol Baxter.

Jon Scott and Denny Gulick
September 2010

Contents

1

Mathscapes—Fractal Scenery

Anne M. Burns
Long Island University, C.W. Post Campus

Introduction

As a mathematician who started out as an art major, I became very excited when the first home computers appeared on the scene in the mid eighties. In 1987 I attended a conference at NYU (I believe it was called "Computer Graphics for the Arts and Sciences"), and marveled at the beautiful fractal pictures created using mathematical methods. On my way home I stopped at Barnes and Noble and bought *The Beauty of Fractals* by Peitgen and Richter. I was hooked! I bought an IBM PC that had a screen resolution of 320×200 and 3 colors. I learned about recursive functions and spent hours drawing stick figures of trees and other fractal figures. Now my PC has a resolution of 1600×1280 and 16,777,216 possible colors!

Learning how to create fractal scenery is a wonderful way for students to learn some mathematics and some elementary computer programming. I will introduce some simple "data structures" used in computer science. Much of my inspiration came from the late eighties and early nineties editions of the journal *Computer Graphics*, a publication of SIGGRAPH (the Special Interest Group on Graphics) of the ACM. The graphics in that journal were produced on large computers using very sophisticated and complicated rendering techniques that can only be accomplished by highly skilled programmers using super computers. They may also require a good knowledge of botany and the laws that govern the birth of inflorescences, or the bending of branches under the force of gravity. My goal has been to distill the underlying mathematical ideas from these papers and to cast them in a way that can be understood by undergraduate math majors. Using probability, linear algebra, geometry and trigonometry, we can teach a student how to investigate an object found in nature; he or she can ponder the question of how a tiny seed holds all the instructions for its growth. Finally it teaches the importance of parameters in modeling. I have tried to simplify the programming and use mathematics that an undergraduate math major can understand but still produce fractal landscapes that are greeted with "awesome" when displayed.

I will start with plant models. Two-dimensional models are easy to understand and are an excellent way to teach recursion. There are several interesting ways to model plant growth and it gives us a chance to learn a little bit about algorithmic processes and the use of parameters. Next we examine an algorithm that can be used to model both fractal clouds and fractal mountains depending on the interpretation of the numbers generated. Finally we will put it all together into a fractal scene such as the one in Figure 1.1, "Fractal Mountains".

Section 1.1 will deal with the modeling of some common inflorescences using simple recursion and how we can

Figure 1.1. "Fractal Mountains", 2005

use mathematics to manipulate nature's rules and produce exotic looking imaginary flowers. In Sections 1.2–1.5 we will examine different algorithms for modeling branching structures like trees and other plants, which we will then embellish with the flowers that we produced in Section 1.1.

1.1 Modeling Inflorescences with Simple Recursion

A number of years ago I figured out how to write a computer program that would draw a simple stick figure fractal tree on my first IBM PC. The idea that a simple five line function could produce a geometric object of such complexity was a revelation to me. Computer graphics has made it possible to model plant growth and even to invent exotic imaginary plants by observing the patterns in nature and compounding them.

A simple technique in mathematics called *recursion* can be implemented easily in a computer program. A recursive function is a function that calls itself, so a recursive function allows us to repeat a simple replacement rule over and over again, changing parameters at each stage. This is apparently what happens in nature; each seed of a plant contains all the instructions necessary to tell that plant when to branch, when to flower and when to slow its growth rate. Plant structures illustrate what fractal geometers call *self-similarity*. A branch of a tree looks like a small replica of the tree itself and even the vein structure of a leaf often looks like a two dimensional picture of the tree it came from. The technique of recursion allows us to generate patterns of enormous complexity by just specifying a few simple directions, and then letting the program take over. For example, suppose we start with a single line segment and a simple replacement rule which says: at each time stage, replace each line segment in the current figure by five new line segments as in Figure 1.2. After several stages it bears some resemblance to a real plant.

When implementing recursion on a computer we have to tell the function when to stop; so one of the parameters of the recursive function will be the number of stages to draw. If we let n be a positive integer greater than 1, an algorithm for the stage n plant in Figure 1.2 might look like the following:

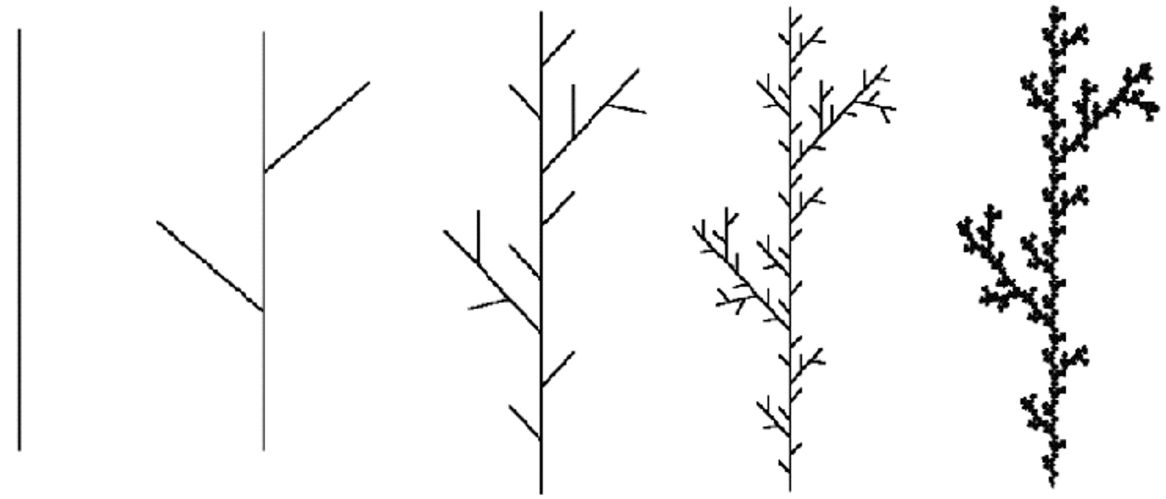


Figure 1.2.

The arguments of the function will be n (the number of stages), θ (the current drawing angle, which is the angle between the line segment and the horizontal), (x, y) (the current point at which to begin a line segment), and l, (the length of the line segment to be drawn).

We'll call our function *Plant*(n, θ, x, y, l). The body of the function will look like this:

1. *Plant*(n, θ, x, y, l);
2. If $n = 0$ then draw a line segment at (x, y), at an angle θ with the horizontal, and with length l
3. Else:
4. $l = l/3$
5. *Plant*$(n - 1, \theta, x, y, l)$
6. $x = x + l * \cos(\theta), y = y + l * \sin(\theta)$
7. *Plant*$(n - 1, \theta + \pi/3, x, y, l)$
8. *Plant*$(n - 1, \theta, x, y, l)$
9. $x = x + l * \cos(\theta), y = y + l * \sin(\theta)$
10. *Plant*$(n - 1, \theta - \pi/3, x, y, l)$
11. *Plant*$(n - 1, \theta, x, y, l)$
12. End (else)
13. End (function)

To see how this works, let us trace through the algorithm for $n = 1$, $\theta = \pi/2$, $x = 0$, $y = 0$ and $l = 1$. In the program we would call *Plant*$(1, \pi/2, 0, 0, 1)$. This is what happens: Line 2: Since $n \neq 0$, the "If $n = 0$" clause is false, and we proceed to the "Else", line 3. Since $l = 1$ the variable l becomes $1/3$ (line 4), and (line 5) the function calls itself, now with the stage $n - 1 = 0$ and $l = 1/3$, *Plant*$(0, \pi/2, 0, 0, 1/3)$. Going back to the beginning of the function with the new parameters, this time $n = 0$, so the function draws a line segment at an angle $\pi/2$ starting at $(0, 0)$ of length $1/3$. Now the function (in its first call with the original parameters) has finished with line 5, so it goes on to line 6. x becomes $0 + (1/3)\cos(\pi/2)$, y becomes $0 + (1/3)\sin(\pi/2)$, that is, x remains 0 but y becomes $1/3$ (so we have drawn the bottom vertical line segment in Figure 1.2b, and our current position is now at $(0, 1/3)$). In line 7 the function is called again, *Plant*$(0, \pi/2 + \pi/3, 0, 1/3, 1/3)$; this call draws the lowest branch left of the main "trunk". Line 8 draws the next vertical segment; line 9 moves the current point to $(0, 2/3)$ and line 10 draws the

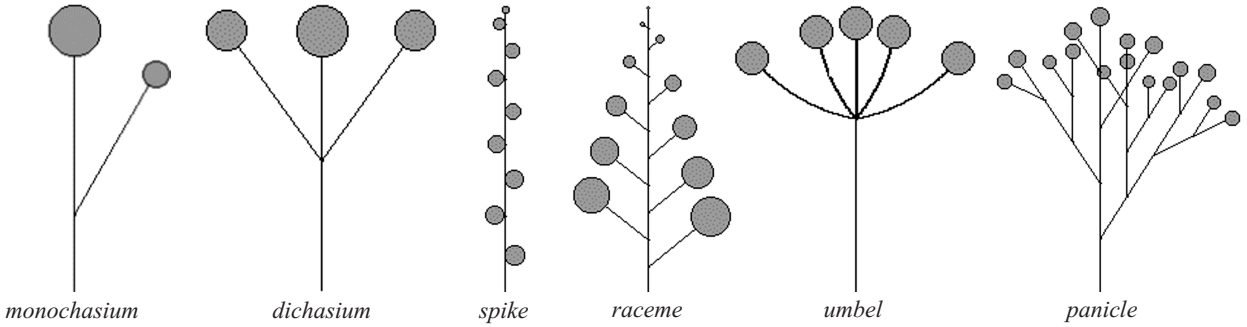


Figure 1.3. Some common inflorescences

segment to the right of the main "trunk". Line 11 draws the final vertical segment. Line 12 ends the else clause and Line 13 ends the function.

The reader should try tracing through this algorithm starting with $n = 2$ and using Figure 1.2c as a guide.

C.L. Porter's *Taxonomy of Flowering Plants* [9] contains diagrams of common inflorescences (flowering structures) and compound inflorescences: stick figures with small disks representing the individual flowers. The diagrams of the compound inflorescences looked similar to my computer graphic stick figures; I realized that using recursion I could not only reproduce them but could carry out the recursion to any desired number of stages. I start, as Porter did, with a simple initial figure, a line segment topped with a disk. Figure 1.3 shows some of the common inflorescences. For each inflorescence a simple mathematical rule for compounding can be detected by looking at the diagrams.

Figure 1.4 shows the compounding of some of the inflorescences in Figure 1.3.

Let us define a few terms. A *peduncle* is the stalk of an inflorescence. "A *monochasium* is a peduncle bearing a terminal flower and, below it, one branch that produces a single lateral flower. The terminal flower is older. This is a *simple monochasium*. A repetition of this on the lateral branches produces a *compound monochasium*." [9].

"A *dichasium* is a peduncle bearing a terminal flower and a pair of branches that produce lateral flowers. The oldest flower is the central one. This *simple dichasium* is a common unit making up parts of many more complex inflorescences. A repetition of this on a lateral pair of branches produces a *compound dichasium*." [9]. In the compound monochasium and dichasium we have rendered older flowers by larger discs.

"An *umbel* is an inflorescence having several branches arising from a common point at the summit of the peduncle. If these branches end in flowers we have a *simple umbel*; if they end in secondary *umbellets* we have a *compound umbel*." [9].

"A *panicle* is a more or less elongated inflorescence with a central axis along which there are branches which are themselves branched. These may be a sequence of blooming from the base upward, but some panicles are made up wholly of dichasia." [9]

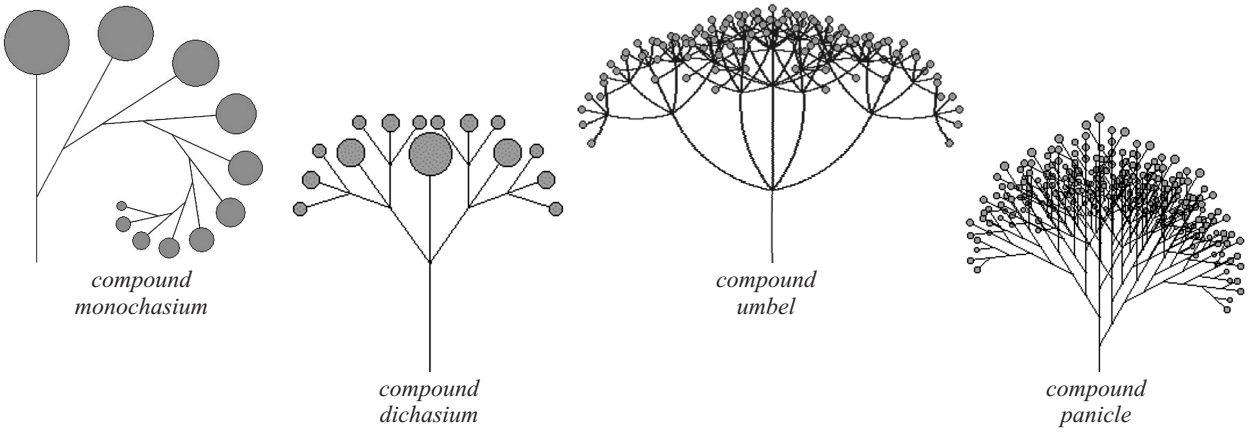


Figure 1.4. Compound inflorescences modeled using simple recursion

Figure 1.5. Some imaginary inflorescences

If we add to our function more parameters such as length, thickness and curvature of the line segments, we can create more realistic looking inflorescences or invent our own. We can also use random numbers and probabilities in a program to simulate the effects of external forces such as the amount of light, temperature and nutrients a plant receives; then the branches of our computer-generated plant and the branching angles are not identical as they are in the "plant" in Figure 1.2. Using random numbers to vary line lengths and angles, replacing lines with curves, and varying the size and color of the disks in the inflorescences allows us to imitate nature, or to invent our own creations. Figure 1.5 shows some imaginary inflorescences created using simple recursive procedures.

1.2 String Rewriting and L-Systems

Aristid Lindenmayer, a biologist from the University of Utrecht, conceived of a way to model the structure and growth of plants by generating new character strings from existing ones using ideas from formal language theory. Premislaw Prusinciewicz, in his beautiful book [10], is an excellent source. Students find the idea that a string of characters can be interpreted as a plant to be an amazing example of modeling. We will start with how to generate long strings of characters from an initial character, and then show how to interpret the string of characters as plants.

First we use a set of rewriting rules called *productions* to successively replace each character in a string of characters

by another string of characters at each stage in the development of the string. Start with the simplest case, called *context-free L-Systems* (a formal definition can be found in [11, 12, 13]): We start with an initial string which we call the *axiom*, and a set of rules which we call *productions*. Each production has a left side and a right side. For the simplest case we will make the left side of each production a single character and the right side a string of characters. The rewriting takes place in discrete stages. The initial string will consist of a single character, the *axiom*. To get from one stage to the next in the string's development we will traverse the current string of characters from left to right. Whenever we encounter a character that is the left side of a production, we replace that character by the right side of that production. By convention, when we encounter a character that is not the left side of any production we replace that character by itself.

Example 1.1. The *alphabet* will consist of the set of characters {I, [,] , (,)} and the axiom will be the character I. Our L-System will have one production: I → I[I]I(I)I This means that we will start (at stage 0) with the single character I. At stage 1 we have replaced the I with I[I]I(I)I. At stage 2 we replace each I in the stage 1 string with I[I]I(I)I. This is what our string will look like after one and two stages:

Axiom: I

Stage 1: I[I]I(I)I, Stage 2: I[I]I(I)I [I[I]I(I)I] I[I]I(I)I (I[I]I(I)I) I[I]I(I)I

At stage 3 the string will consist of 249 characters!

The leftmost figure in Figure 1.6 will be an interpretation of the first stage in this example. In this example the only characters are I, [,], (, and). To get to the geometry of the plant from the generated string, each geometric part of the plant is assigned a character. For this simple example, the only geometric object is a line segment. We will let the character I represent a line segment (in botany, an internode, or branch). The brackets [ ] are used to enclose a branch to the left and parentheses ( ) are used to enclose a branch to the right. The string of characters in our example at stage 1 can be used to describe the leftmost imaginary plant in Figure 1.6. Interpreting the string I[I]I(I)I, we traverse the string from left to right, and the "plant" will grow from the base upwards. The first character, I, will be the bottom vertical line segment. Next we encounter a [which tells us to turn left (or increment our current drawing angle). The next I tells us to draw another line segment at the new angle. Then we come to the character]. This tells us to end the branch and go back to where the branch began and retrieve the (x, y) coordinates and the original drawing angle. The next I says draw a vertical segment and then the next three characters (I) are interpreted as the branch to the right. Finally, the last I is the line segment at the top of the plant.

The astute reader will notice that this "plant" is the same one as the one in Figure 1.2, and can easily be drawn using recursion. However, by introducing more symbols and more productions, we will be able to draw a large variety of plant structures.

Writing functions that draw the lines, leaves and flowers gives students the opportunity to use some elementary trigonometry and algebra. Saving the current (x, y) position and the current angle when branching is a good application of implementing a *stack* in Computer Science.

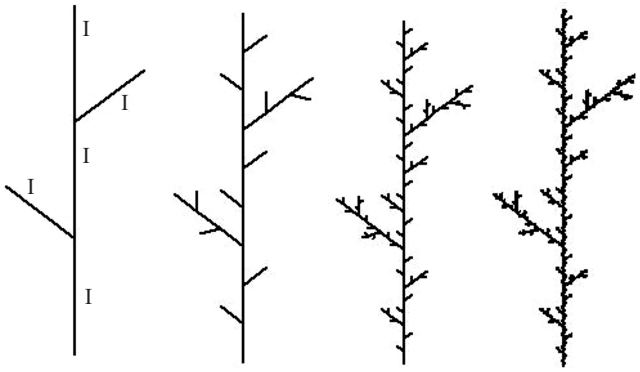
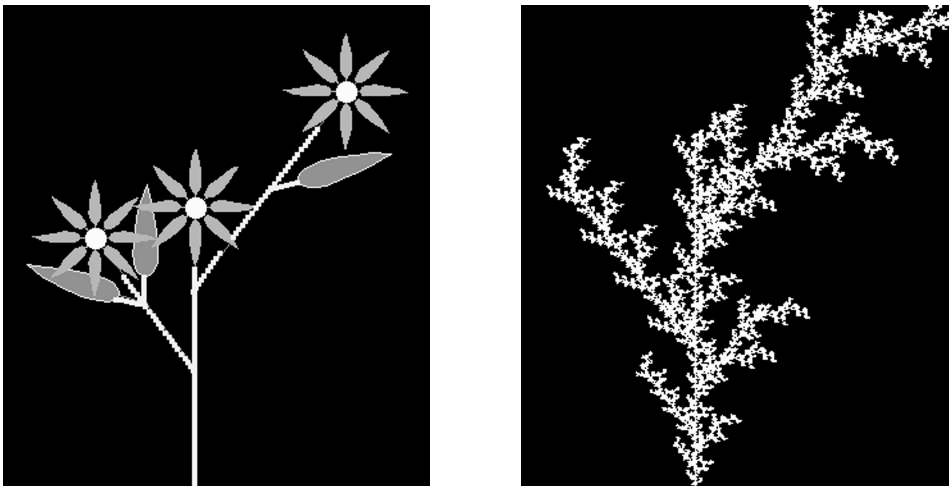


Figure 1.6. Stages 1–4 of Example 1.1



(a) (b)

Figure 1.7.

The plant in Figure 1.2 is not that different from the plant in Example 1.1. Two new characters have been added to the alphabet, “L” which we interpret as a “leaf” and “F” which we interpret as a “flower”.

I again represents an internode (or branch), L a leaf, F a flower, brackets [,], are used to enclose a branch to the left and parentheses (,) are used to enclose a branch to the right. The string of characters I[I[L](L)IF]I(II(L)IF)IF can be used to describe the imaginary plant in Figure 1.7(a). In Figure 1.7(b) we see a later stage in the development of this “plant”. You can see the self-similarity and fractal nature of the generated plant.

In Figure 1.8 we have modeled opposite branching and added some fanciful flowers to make a decorative plant.

Figure 1.8. A decorative plant

String rewriting can be used to model the common inflorescences (see previous section). Using the symbols I (to represent a line segment), [ ] (enclose a branch to the left), ( ) (enclose a branch to the right), and F (to represent a flower), Figures 1.9–1.12 give the rules for generating a monochasium, a dichasium, an umbel and a panicle (see Section 1.1).

Figure 1.9. *Rules for a monchasium*: I → I, A → I(A)B, B → B, Axiom: A

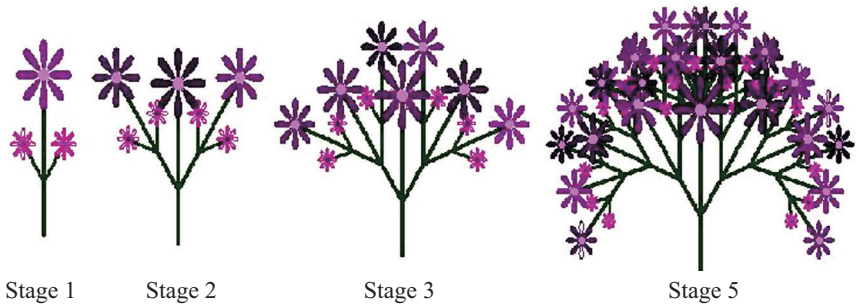


Figure 1.10. *Rules for a dichasium*: A→ I[A](A)B, Axiom: A

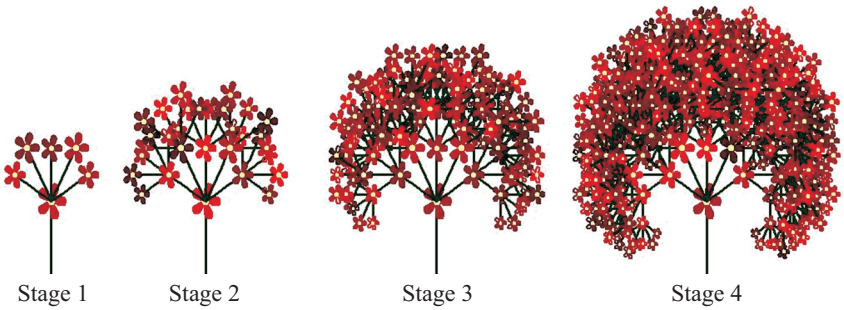


Figure 1.11. *Rules for an umbel*: I → ((IF))(IF)[(IF)][IF][[IF]], Axiom: I

Figure 1.12. *Rules for a panicle*: I → A[IF]A(IF)IF, Axiom: I

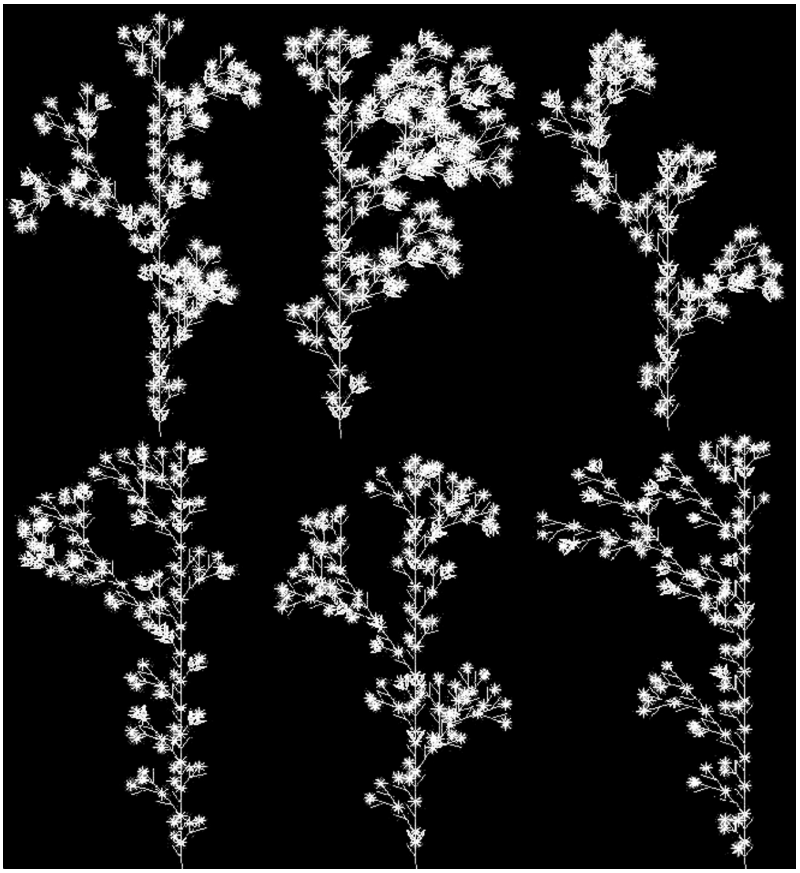


Figure 1.13. A stochastic L-System

There are a number of ways to make our plant growth less predictable so that our plants look more realistic and more closely mimic the growth of real plants which are subject to variations in exposure to light, wind, and other forces of nature. One way to do this is a *Stochastic L-System* in which two or more productions have the same left side but different right sides. Suppose we have n productions with the same left side but different right sides. We assign to each production a positive real number p_j such that $p_1 + p_2 + \cdots + p_n = 1$. At each stage in the processing of the string we generate a random number r. If $r < p_1$ we replace I with the right side of the first production. If $p_1 \leq r \leq p_1 + p_2$ we replace I with the right side of the second production, etc. Figure 1.13 shows six "plants" generated by a stochastic L-System consisting of three productions with left sides I and right sides: (1) I $\rightarrow$ I[I]IF, $p_1 = .23$, (2) I $\rightarrow$ I(I)[AF](I)IF, $p_2 = .5$ and (3) I $\rightarrow$ I[L](L)AF, $p_3 = .27$. Notice that the plants look like they belong to the same "family", but they are not identical.

L-Systems also allow us to model various stages, such as a flowering sequence, in the growth of a plant. We keep track of the stage as we go through the series of replacements of the strings with their next stage strings by using a variable, say t, to represent the time stages in the plant's growth. For a given left side of a production we can have more than one right side; the particular one chosen will depend on the value of t. To illustrate this idea, here is a simplified version of a flowering sequence where the flowers mature earlier on the lower branches. In this example the axiom is I and we let t represent the stage; that is at $t = 0$ the string consists of the axiom I.

The productions are:

$$\begin{aligned}
&\mathrm{I} \rightarrow \mathrm{A[B]I} \quad \text{for } t \leq 2\\
&\mathrm{I} \rightarrow \mathrm{AB} \quad \text{for } t = 3\\
&\mathrm{B} \rightarrow \mathrm{A}\, f(0)\\
&f(n) \rightarrow f(n+1) \quad \text{for } n \leq 2\\
&f(3) \rightarrow \varepsilon \ \text{(the empty string)}
\end{aligned}$$

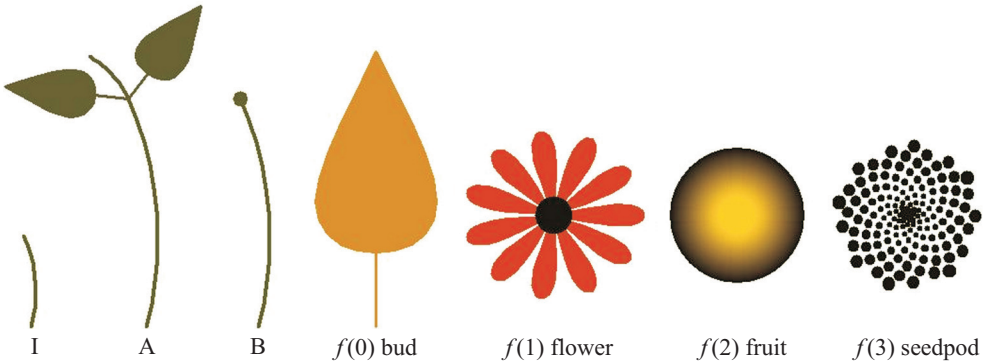


Figure 1.14. Representation of string characters for a parametric L-System

Figure 1.14 shows the representation of each of the characters in the string. (If you are a teacher, note that writing the routines that draw each of the figures involves a little math.)

Figure 1.15 illustrates the flowering sequence and life cycle of the "reproductive organs" of the plant.

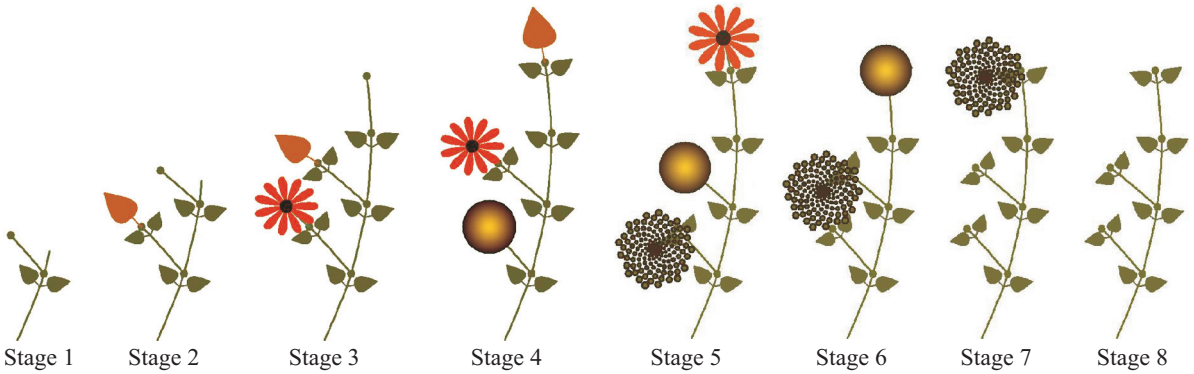


Figure 1.15. 8 stages in the parametric L-system

Up until now we have considered only productions with a single character on the left side. These are called *context-free*. *Context-sensitive* rewriting rules give us even more flexibility for propagating signals through a plant. Here the decision to replace a character by a new string is determined not only by the character itself, but also by the character to its left (*left context-sensitive*) and/or the character to its right (*right context-sensitive*).

In a context-sensitive L-System a production can be written $A < B < C \rightarrow D$ which is interpreted as: replace B by D if and only if B is preceded by A and succeeded by C. An exercise for students might be to show that a signal may be propagated from the root to the tips of a plant using a left context-sensitive system, and from the tip of the plant to the root using a right-context sensitive system. Here is an exercise:

> Given an initial tree represented by SI[I[A](A)A](I[A](A)A)I[A](A)A and the productions S $<$ I $\rightarrow$ IS, S $<$ A $\rightarrow$ IF, S $\rightarrow$ ε (where ε represents the "empty" string). Let I and A be represented by line segments, (the symbol S is not part of the plant's geometry) and draw a plant represented by the original string and after one, two and three stages. (The symbol S should have disappeared after three stages.)

String rewriting is an excellent topic for generating interest in mathematics and computer programming. I have presented only a very simple introduction to this exciting topic. It is easy to see how other fractals can be generated using string rewriting. To learn more about this topic I highly recommend Prusinkiewicz' beautiful book [10].

1.3 DeReffye's Method (for want of a better name)

I call this method deReffye's Method, since I first encountered it in the SIGGRAPH Proceedings (1988) in an article based on a master's thesis by Philippe deReffye [7]. We will start with a very simplified version, and it will probably be obvious to the reader how to improve on it.

The basic idea behind this algorithm is as follows: we are going to create a tree from scratch, starting with a single "node". Each node in the tree will be assigned a positive integer, which will be called the *order* of the node. We will create the tree in discrete time steps, $t_1, t_2, \ldots, t_n$. The order of the original node will be 1. At each time step a node in the tree can do one of three things: (i) ramify (create two child nodes), (ii) flower and die (that is, it can't produce any more nodes) or (iii) "sleep" for one time period. If the order of a node is n, the order of its child nodes will be $n + 1$.

In the simple version we will assign three positive real numbers, p_1, p_2, p_3 whose sum is 1. These numbers will represent the probabilities that a node will ramify, flower (and die), or sleep. Later, to more closely mimic nature, we can make these numbers functions of time.

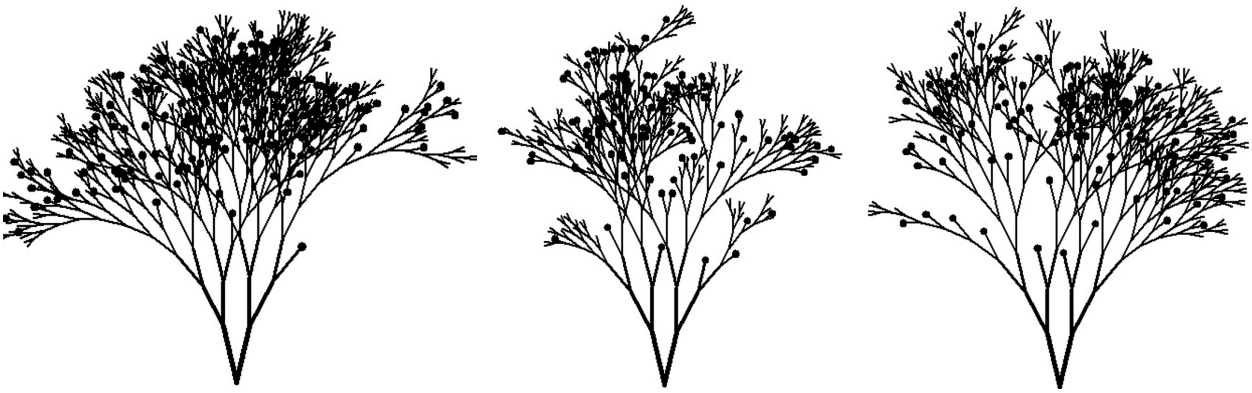


Figure 1.16. Three binary trees generated with the same program; $p_1 = .5$, $p_2 = .15$, $p_3 = .35$. Nodes are connected by lines, flowers are represented by disks. $\theta_0 = \pi/2$.

When a new node is created, it will be assigned an integer representing its order, its (x, y) coordinates and an angle θ, which will determine the angle in the plane at which the line connecting it to its child nodes will be drawn.

Before entering our first time period we will create the root node, assigning it order 1, (x_0, y_0) and θ_0, and we will initialize p_1, p_2, p_3 and a variable representing the length of the line segment that will connect the node with its child nodes. In Figure 1.16 the length of the line decreases as the order of the node increases. We then enter a loop where each pass through the loop represents a tick of the clock (time period). At each pass we examine the list of nodes that currently exist. For each node we generate a random number r between 0 and 1. If $r < p_1$ (the node ramifies) we create two new child nodes. For each child node we assign: its order (the order of its parent plus 1), and its (x, y, θ) coordinates (based on the (x, y, θ) values of its parent). After creating the child nodes and connecting them to their parent nodes on the screen, we delete the parent node from the list. If $p_1 < r < p_1 + p_2$, we draw a disc (flower) , and finally if $p_1 + p_2 < r \leq 1$, the node sleeps, so we just continue on to the next node.

What are some ways to make our trees more varied? Looking at plants in nature we notice that they do not always ramify into two new branches. So we could introduce a *ramification number* which would specify the number of new branches emanating from a node. Figure 1.17 shows a tree with ramification number 3 and the disks replaced by "flowers".

Another thing we can do to make our trees conform more closely to nature as well as more attractive artistically is to make the probabilities p_1, p_2 and p_3 vary with time, so that the probability of ramifying is higher at earlier time periods and the probability of dying (flowering) is higher at a later stage. If we want to end up with a flower at the end of every branch when our picture is finished, we specify a maximum time period $t_{\max}$ and at the stage with $t_{\max}$ we assign $p_2 = 1.0$.

1.4 Modeling Trees with a Stochastic Matrix

Another method of modeling plant growth uses the idea that a plant's geometry can be described using a stochastic matrix. This method first appeared in [15]. This method requires a little probability and the concept of a binary tree. This is an excellent topic for a mathematics class for liberal arts students or any survey course or as a student research project.

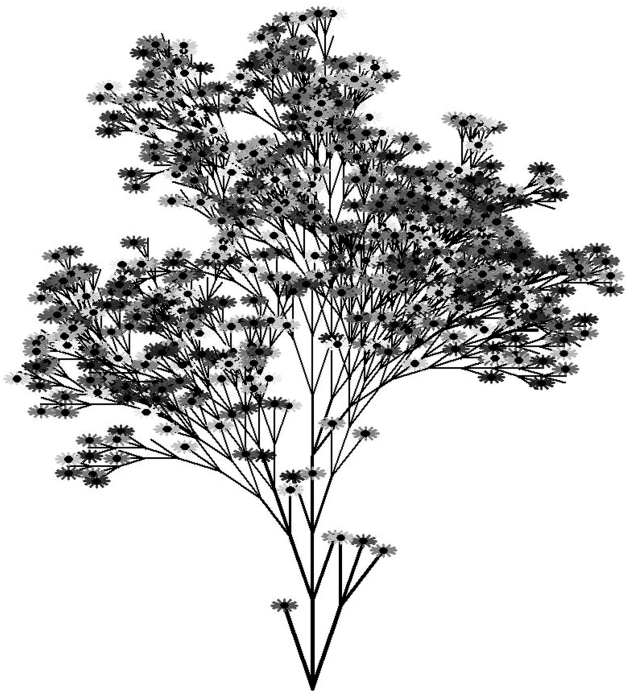


Figure 1.17. Ramification number = 3 and probability of flowering increases with time.

A formal definition of a binary tree can be found in most elementary books on computer science or data structures. For our purposes we can think of a binary tree as a set of nodes that are joined by edges. A node can have 0 or 2 child nodes and each node except for the root node has one parent node. The root node has no parent node. A node with no child nodes is a leaf or terminal node. Figure 1.18 shows a binary tree with 11 leaf nodes.

A stochastic matrix is a square matrix in which the sum of entries in each row is 1:

$$\begin{pmatrix} 1/3 & 1/3 & 1/3 \\ 1/4 & 0 & 3/4 \\ 1/7 & 4/7 & 2/7 \end{pmatrix}$$

(Think of the entries in each row as probabilities.)

For this model we are going to define the order of a node in a binary tree in a non-standard way. For each node n in a binary tree:

$$\text{Order}\,(n) = \begin{cases} 1 & \text{if } n \text{ is a leaf (terminal node)} \\ i & \text{if its two child nodes are either both order } i-1, \text{ or order } i \text{ and } j, \text{ with } j < i. \end{cases}$$

Example 1.2. Let's calculate the order of the nodes in the binary tree in Figure 1.18. Start with the leaf nodes (label them 1) and work down. Label each leaf node 1, then any node whose two child nodes are labeled 1 should be labeled 2, etc. The result of labeling the nodes is pictured in Figure 1.19.

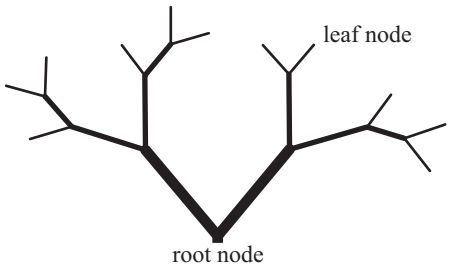


Figure 1.18. A binary tree with 11 leaf nodes

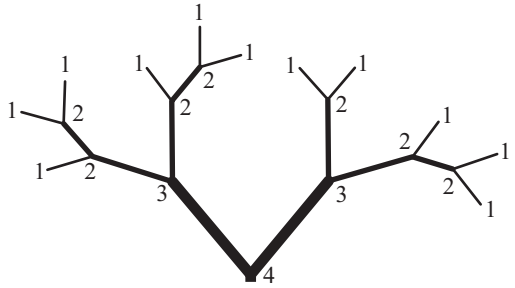


Figure 1.19. The order of nodes in a binary tree.

We are going to work backwards. Our goal is to create a matrix that will describe a family of trees and use that matrix to draw the tree, but first we will see how to create the matrix from a given binary tree. This will give us some insight into what properties a matrix should have to model a given family of trees.

Now we will construct a *ramification matrix* (a_{ij}) for an existing binary tree. Let n_i = the number of nodes of order i, and let m_{ij} = the number of nodes of order i that have child nodes of order i and j with $j < i$, and m_{ii} = the number of nodes of order i that have two nodes of order $i-1$. Now let $a_{ij} = m_{ij}/n_i$.

To illustrate the process, we will use the tree in Figure 1.20 where we have added a trunk below the root node to make it look more like a tree.

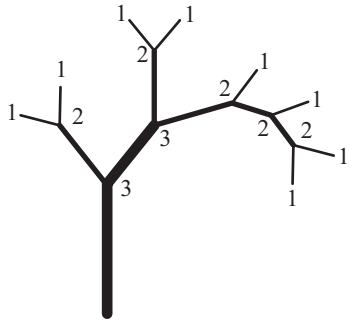


Figure 1.20.

First, draw the desired binary tree and label the order of the nodes. Since the root node has order 3, our matrix will be 3 by 3. We will construct the matrix starting with the bottom row and work our way up.

Counting the number of nodes of order 3 we see that $n_3 = 2$. Since neither of these nodes has children of order 1, $m_{31} = 0$. Since one of them has children of orders 3 and 2, $m_{32} = 1$ and one has children of order 2 and 2, so $m_{33} = 1$. Thus $a_{31} = 0$, $a_{32} = 1/2$ and $a_{33} = 1/2$. Next, to construct Row 2: there are five nodes of order 2, two of them have children of orders 2 and 1, and three of them have children both of order one. Thus $a_{21} = 2/5$ and $a_{22} = 3/5$. The first row will not be used as nodes of order 1 have no child nodes. (Strictly speaking, this is not a stochastic matrix since the first row is all zeros. However the entries in all rows except the first will sum to 1.) So the ramification matrix for this binary tree is

$$\begin{pmatrix} 0 & 0 & 0 \\ 2/5 & 3/5 & 0 \\ 0 & 1/2 & 1/2 \end{pmatrix}$$

There are many interesting classroom activities concerning these matrices, such as what properties must a ramification matrix have, is it always possible that a matrix with these properties be a ramification matrix of a binary tree, what is the ramification matrix of a perfect binary tree (all non-leaf nodes have two children)? An interesting classroom exercise is to take an existing ramification matrix such as the above one and see how many binary trees can be constructed with that matrix.

Now we want to see how to generate a binary tree from a matrix with the required properties. We are going to interpret the entries as probabilities. We will build the tree from the root node up. From the size of the matrix we know the order of the root node. (If the matrix is $n \times n$ the root node will have order n.)

We are going to keep a list of records of current nodes and their properties. The record for each node will contain its order and its (x, y) position and a drawing angle (for the root node we might use $\pi/2$ if we want the tree to grow upward.). Initially the list will consist of only the root node, and we will start with the last row. We generate a random number r between 0 and 1. The first time through we have only one node, the root node. If $r < a_{n1}$ then the root node will have children of orders 1 and n; if $a_{n1} < r < a_{n1} + a_{n2}$ the root node will have children of orders 2 and n; in general if

$$\sum_{k=1}^{j-1} a_{nk} < r \leq \sum_{k=1}^{j} a_{nk}, \quad j < n,$$

the root node will have children of orders j and n. If $r > \sum_{k=1}^{n-1} a_{nk}$ then both children of the root node have order $n - 1$. Each time a new node is born we create a record of its order and its position, which can be constructed from the position of its parent node using a little trigonometry. At each pass through the loop we examine the list of the current nodes. For each node in the list, if the current node has order k we use the kth row of the ramification matrix to determine the order of its children just as we did with the root node and the nth row. Then we create the record for each of the two new child nodes, insert those records into the list, connect the new nodes to their parent with a line segment, and delete the parent node from the list. This is an excellent exercise for a class in computer science. It can also be done by a student familiar with *Mathematica* (see [4]).

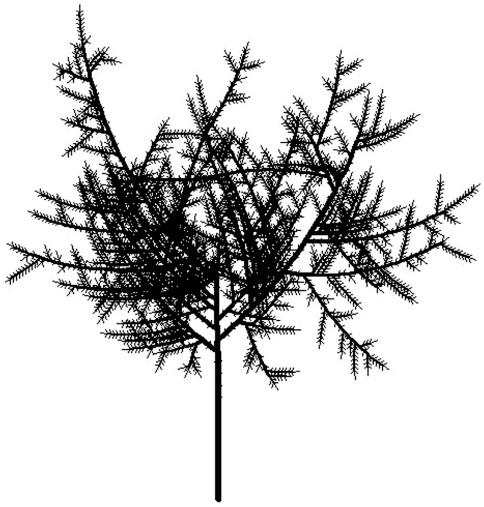


Figure 1.21. For the lower order branches we give a high probability in the first column ensuring a lot of order 1 nodes which we interpret as "pine needles".

It is amazing to see the variety of tree shapes that can be generated by this method. For a bushy tree we can assign equal probabilities to each row (row k will contain k nonzero entries, so let $a_{kj} = 1/k$). To model a conifer we can imagine the needles as terminal nodes (order 1 nodes) and assign a high probability to the entries in the first column as in Figure 1.21. To make the trees more realistic we can vary branch angles, lengths and widths as functions of the order of the nodes from which they emanate.

In Figure 1.22 we see three trees all generated from the same matrix.

1.5 Clouds and Mountains

For the background in our fractal landscapes we turn to an algorithm that can be interpreted in different ways, giving us either clouds or mountains. A rigorous discussion of the mathematics and more variations can be found in [8]. The method is called *random midpoint displacement*. It has some resemblance to a binary search; we bisect the original interval and then bisect the two new intervals, and recursively continue until the intervals are one pixel wide.

We will start by seeing how it works in one dimension. Starting with a horizontal line segment of length $2^n + 1$ pixels for a positive integer n, we assign a height in pixels to the left and right endpoints of the line as in the first frame

Figure 1.22. These three trees are generated from an 8×8 matrix with $a(i,i) = .5$ and $a(i,i-1) = .5$ and all other entries 0.

of Figure 1.23. That is, we label our pixels $x_0, \ldots, x_{2^n}$ and at the beginning only y_0 and y_{2^n} are assigned values. We are going to use a recursive function; at each stage we divide the interval in half and a random number is multiplied by a scale factor and then added to or subtracted from the average of the height at the left endpoint and the height at the right endpoint; the result is assigned to the height of the midpoint. At the first stage we do this for the initial interval. Then we make two calls to the recursive function that repeats the process for the two new intervals. This is another application of recursion. The recursive function takes as its arguments the x coordinate and height of the left and right endpoints of an interval. The stopping point for the recursion is when $x_{\text{left}} = x_{\text{right}}$. Figure 1.23 shows the heights assigned initially, after one, two and three stages of recursion. Figure 1.24 shows the result of carrying out the rule until the left and right endpoints of each interval are the same.

Using shading and layering, even this very simple two-dimensional model can be used for a background in a scene as in Figure 1.25.

For more realistic mountains the midpoint displacement rule illustrated in Figures 1.23 and 1.24 can be extended to a two-dimensional model in which we obtain a height field over a two-dimensional grid. We can use this to model

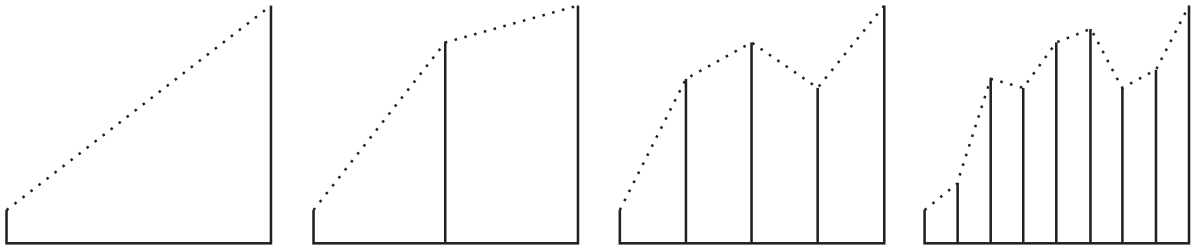


Figure 1.23. A random midpoint replacement rule.

Figure 1.24. Result of replacement rule in Figure 1.23.

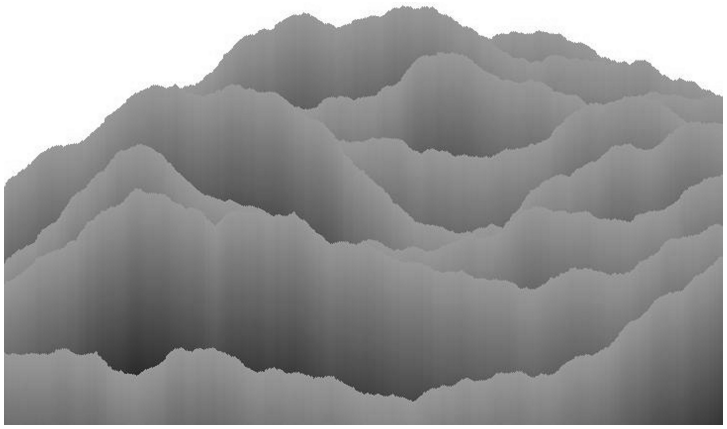


Figure 1.25. Several layers of two-dimensional midpoint replacement "mountains".

both mountains and clouds. To model clouds, we assign a ramp of colors to the heights, while to model mountains we use some trigonometry and calculus to project the three-dimensional height field onto a two-dimensional surface.

First we outline the recursive midpoint algorithm for generating the height field. We start with a $2^n + 1$ by $2^n + 1$ grid of points in the plane, where n could be 9, for example, for a 513 by 513 grid. We assign a number to each of the 4 corner points. At step 1 we assign a value to the point labeled 1 in Figure 1.26a; the value is the average of the numbers at the 4 corner points plus a random number. At step 2 we assign values to the points labeled 2 in Figure 1.26b, by taking the average of the three closest previously assigned points plus a random amount. Figures 1.26c and 1.26d show the order in which the next several points are assigned. At each stage the random amount is scaled down.

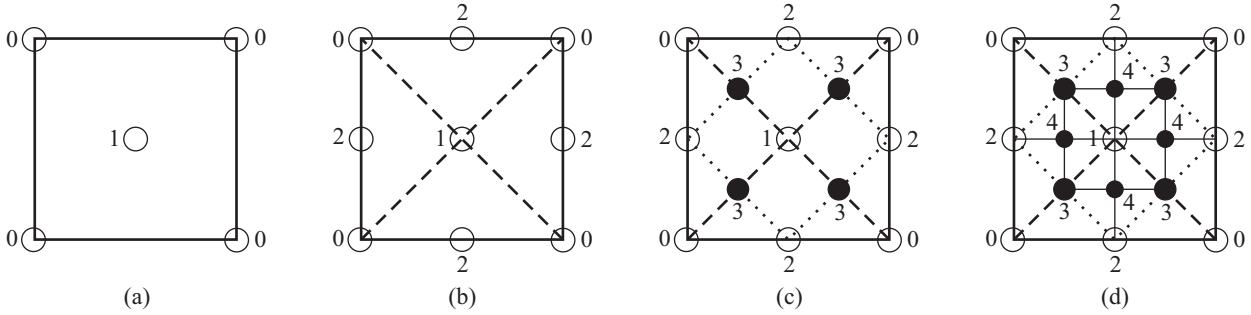
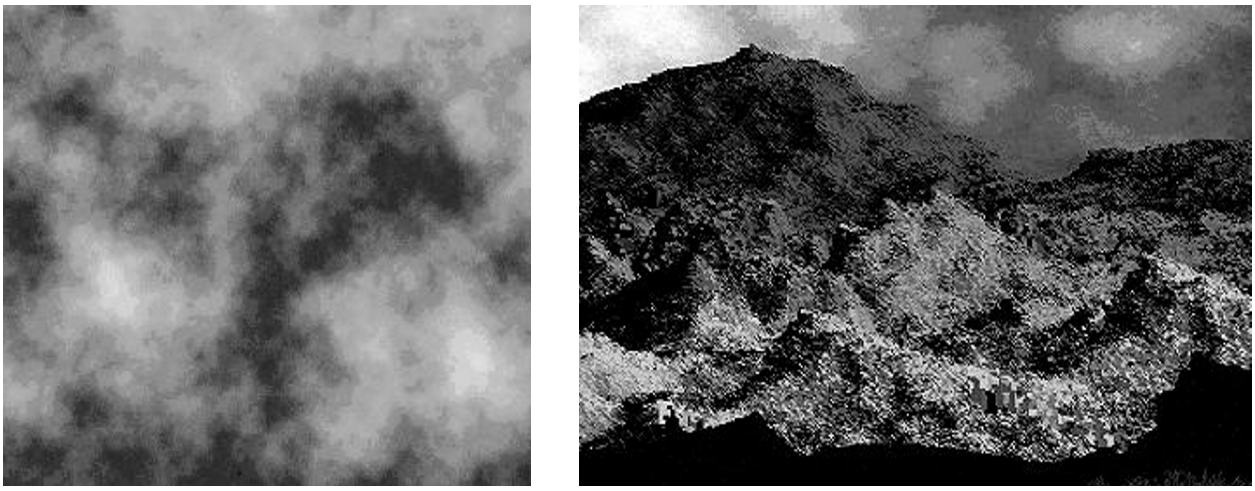


Figure 1.26. Can you guess which pixels will be assigned at Stage 5?

Once the height field is generated we can assign a continuous ramp of colors to the heights and render it as clouds as illustrated in Figure 1.27a, or, using some calculus and trigonometry, as three-dimensional mountains illustrated in Figure 1.27b. By changing the scale factor that we use in scaling down the random amount that we add at each stage, we can produce sharp fractal-like mountains and clouds or soft rounded mountains and cumulus clouds.

Rendering the mountains in three dimensions requires quite a bit more work and offers opportunities for students to use their calculus and trigonometry.

We now have a $2^{n+1} \times 2^{n+1}$ matrix of heights; let i, the row number, represent the x-coordinate, j, the column number, represent the y-coordinate and the matrix entry, $f(i, j)$, represent the height. Define $S_{i,j}$ to be part of the (plane) surface defined by $\big(i, j, f(i, j)\big)$, $\big(i + 1, j, f(i + 1, j)\big)$, $\big(i + 1, j + 1, f(i + 1, j + 1)\big)$ that lies above the square defined by $(i, j, 0)$, $(i + 1, j, 0)$, $(i + 1, j + 1, 0)$ and $(i, j + 1, 0)$. Despite the fact that the surface obtained by patching together these surfaces has small discontinuities, this gives satisfactory visual results; a student project could be to define a continuous, or even differentiable, surface from this matrix. We need to project this three-dimensional surface onto a two-dimensional computer screen. To do this, we specify two unit vectors in polar coordinates, a view vector, **V**, and a light source vector, **L**. We picture the computer screen as a plane surface whose normal is in the direction of the view vector, **V** as in Figure 1.27. We require a unit normal vector to the surface, $S_{i,j}$; the three points,



(a) 2D rendering (b) 3D rendering

Figure 1.27.

$(i, j, f(i, j))$, $(i+1, j, f(i+1, j))$, $(i+1, j+1, f(i+1, j+1))$ determine two vectors, we can use their vector product to find $\mathbf{n}_{i,j}$, the unit normal vector to $S_{i,j}$ (See Figure 1.28).

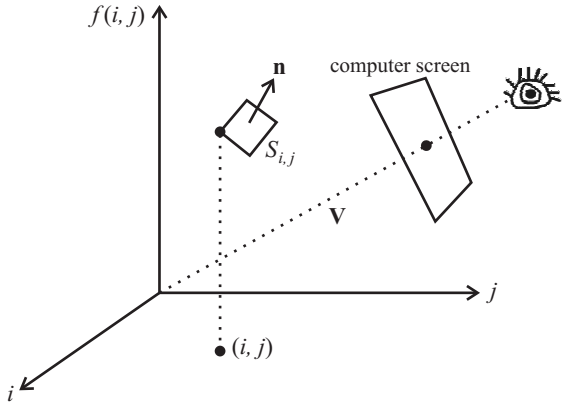


Figure 1.28.

The view vector also allows us to determine if $S_{i,j}$ is visible. Suppose the coordinates of **V** are positive, that is, **V** lies above the first quadrant (otherwise we can adjust the order in which we process the points). We examine the projection of the point $(i, j, f(i, j))$ on the computer screen. If that point has already been rendered it means that the point is hidden from view and we go on to the next point. If the point has not been rendered then we determine if $S_{i,j}$ is visible from the viewpoint **V**; the dot product of $\mathbf{n}_{i,j}$ and **V** will be nonnegative if $S_{i,j}$ is visible. If $S_{i,j}$ is visible the light source allows us to use shading to make the mountains look three-dimensional. For simplicity we assume that the light source is sufficiently far away from the mountains so that we can ignore the small change in angle as we move from one point on the surface to the next. To determine the color and value of the color of this surface, we calculate the dot product of $\mathbf{n}_{i,j}$ with the unit vector in the direction of the light source, **L**. If the dot product is 1, then the light is hitting the surface directly and we color the projection of $S_{i,j}$, the lightest and brightest color; if the dot product is not positive then no light is hitting that surface and we color it our darkest color. If the dot product is between 1 and 0 we choose a color from a ramp of colors from light and bright to dark and dull. Figure 1.29 shows a line drawing of "mountains" generated by the two-dimensional midpoint algorithm. In this figure the points $(i, j, f(i, j))$ have been projected onto the computer screen using the view vector **V** and connected by lines, but the surfaces have not been rendered. A good project for students would be to devise a way to obtain a smoother surface from the matrix of heights. For more details see [5] and [8].

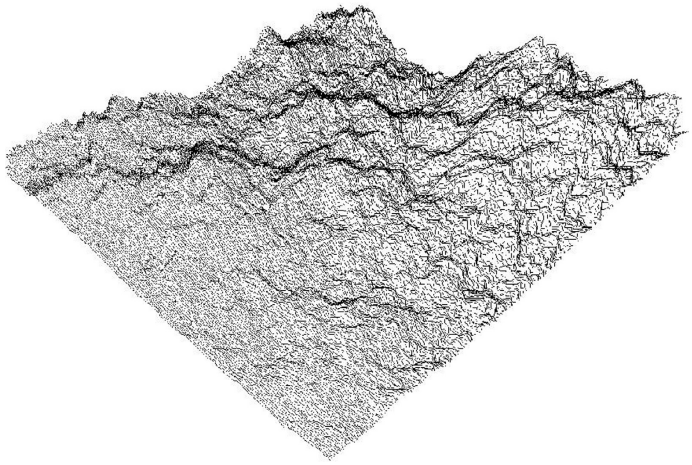


Figure 1.29. "Mountains" generated by the midpoint algorithm.

1.6 Conclusion—Putting It All Together

After writing computer programs that model the various components of the scene it is time for the fun part, making a *mathscape*. You can plan or sketch an outline on paper, or you can just experiment and let your picture evolve. Even if you do make a sketch before starting, it is unlikely that your scene will bear much resemblance to your original plan. The exciting thing is the unpredictability and the happy accidents that occur. And once you have the routines in place it is a simple matter to erase your picture, observe the effect of changing a few parameters, save what you like and discard what you don't like. You do not have to worry that you don't have the imagination to make a picture; the whole process is like a fractal itself: each time you change a parameter or a line in your program another whole set of ideas emerges. The following are some mathscapes produced by this author.


Anne Burns, 2007


Anne Burns, 2005


Anne Burns, 2007


Anne Burns, 2006

Acknowledgements

Some of the ideas and figures in this chapter have appeared in two volumes of the Proceedings of conferences of the organization *Bridges: Mathematical Connections in Art, Music and Science*, specifically *Renaissance Banff* [6] and *Bridges Donostia* [2]. I would like to thank the co-editor of both volumes, Reza Sarhangi, for permission to reproduce several of the figures. Some of the figures in Sections 1.1 and 1.5 were first published in [6], and some of the material and figures in Section 1.2 first appeared in [2].

Bibliography

[1] Burns, Anne M., The Flowering of String Rewriting Systems, *The College Mathematics Journal*, 23(3), (May 1992), 225–236.

[2] ——, Imaginary Gardens: A Model for Imitating Plant Growth, *Proceedings of Bridges Donostia 2007*, Edited by Reza Sarhangi and Javier Barrallo.

[3] ——, Recursion: Real and Imaginary Inflorescences, *The UMAP Journal*, 21(4), (Winter 2000).

[4] ——, Modeling Trees with a Stochastic Matrix, *The College Mathematics Journal*, 29(3), (May 1998), 230–236.

[5] ——, Making Mountains from a Sum of Molehills, *The College Mathematics Journal*, 26(1), (January 1995), 51–57.

[6] ——, Recursion in Nature, Mathematics and Art, *Proceedings of Bridges Renaissance Banff*, Reza Sarhangi and Robert V. Moody (Eds.), 2005.

[7] de Reffye, Philippe, Claude Edelin, Jean-Françon, Marc Jaeger, and Claude Puech, Plant Models Faithful to Botanical Structure and Development, *Computer Graphics*, 22, 4 (August 1988).

[8] Peitgen, Heinz-Otto and Dietmar Saupe, Editors, *The Science of Fractal Images*, Springer-Verlag, 1988. Chapter 1 Fractals in Nature: From characterization to simulation by Richard Voss (for the random midpoint displacement method).

[9] Porter, C.L. Porter, *Taxonomy of Flowering Plants*, W.H. Freeman and Co., 1959.

[10] Prusinkiewicz, Przemyslaw and Aristid Lindenmayer, *The Algorithmic Beauty of Plants*, Springer-Verlag, 1990.

[11] Prusinkiewicz, Przemyslaw and James Hanan, *Lindenmayer Systems, Fractals, and Plants*, Lecture Notes in Biomathematics, Springer-Verlag, 1980.

[12] Rozenberg, G. and A. Salomaa (Eds.), *Lindenmayer Systems*, Springer-Verlag, 1992.

[13] —— (Eds.), *The Book of L*, Springer-Verlag, 1986.

[14] Salomaa, Arto , *Formal Languages*, Academic Press, 1973.

[15] Smith, Alvy Ray, Plants, Fractals and Formal Languages, *Computer Graphics*, 19(3), (July 1984).

[16] Viennot, Xavier Gerard, Georges Eyrolles, Nicholas Janey, and Didier Argues, Combinatorial Analysis of Ramified Patterns and Computer Imagery of Trees, *Computer Graphics*, 23(3), (July 1989).

[17] en.wikipedia.org/wiki/L-_system.

2

Chaos, Fractals, and Tom Stoppard's *Arcadia*

Robert L. Devaney
Boston University

Tom Stoppard's wonderful play, *Arcadia*, offers teachers of both mathematics and the humanities the opportunity to join forces in a unique and rewarding way. The play features not one but two mathematicians, and the mathematical ideas they are involved with form one of the main subthemes of the play. Such contemporary topics as chaos and fractals form an integral part of the plot, and even Fermat's Last Theorem and the Second Law of Thermodynamics play important roles.

The play is set in two time periods, the early nineteenth century and the present, in the same room in an English estate, Sidley Park. As the play opens, we meet Thomasina, a young thirteen year old girl who struggles with her algebra and geometry under the watchful eye of her tutor, Septimus Hodge. But Thomasina is not your typical mathematics student; as becomes clear as the play unfolds, she is a prodigy who not only questions the very foundations of her mathematical subjects, but also sets about to change the direction of countless centuries of mathematical thought. In the process, she invents "Thomasina's geometry of irregular forms" (aka fractal geometry), discovers the second law of thermodynamics, and lays the foundation for what is now called chaos theory.

In the modern period, we meet Valentine, a contemporary mathematical biologist who is attempting to understand the rise and fall of grouse populations using iteration. As luck would have it, Valentine is heir to Sidley Park and part of his inheritance is a complete set of game books that go back to Thomasina's time. These books detail the precise number of grouse shot at the estate each year. Gradually, he becomes aware of some of the old mysteries surrounding Sidley Park, including Thomasina's discoveries, and this sets the stage for a unique series of scenes that hop back and forth between the nineteenth century and the present.

At the same time, Valentine's friend, Hannah Jarvis, is attempting to understand some of the mysteries surrounding some of the historical events that occurred around the period that Thomasina lived at Sidley Park. Thomasina had died in a tragic fire the night before her seventeenth birthday, and right about that time, a hermit moved into a cottage on the estate and lived there for many years. As part of her research, Hannah finds out that this hermit had spent his entire life working out what appeared to her to be incomprehensible mathematical equations. With Valentine's help, Hannah comes to realize that the hermit turns out to be Septimus, who, after Thomasina's death, spent the rest of his life trying to push her ideas forward.

Mathematics is not the only theme of this play, but the ideas of regular versus irregular geometry or order versus chaos seem to pervade all of the other events occurring at Sidley Park. We are thrust into a debate about emerging British landscape styles featuring the orderly classical style versus the irregular, "picturesque" style. Valentine and

Hannah methodically proceed to uncover Sidley Park's secrets, in stark contrast to her nemesis, Bernard Nightingale, who jumps from one theory to another with reckless abandon. Indeed, the entire play pits the rationalism of Newton against the romanticism of Lord Byron.

2.1 Thomasina's Geometry of Irregular Forms

Thomasina does not like Euclidean geometry. Early in the play she chides her tutor, Septimus, "Each week I plot your equations dot for dot, x's against y's in all manner of algebraical relation, and every week they draw themselves as commonplace geometry, as if the world of forms were nothing but arcs and angles. God's truth, Septimus, if there is an equation for a bell, then there must be an equation for a bluebell, and if a bluebell, why not a rose?" So she decides to abandon classical Euclidean geometry in order to discover the equation of a leaf.

Years later, Hannah discovers Thomasina's workbooks in which she has written, "I, Thomasina Coverly, have found a truly wonderful method whereby all the forms of nature must give up their numerical secrets and draw themselves through number alone." Hannah asks Valentine how she does this. Val explains that she uses "an iterated algorithm."

"What's that?" Hannah inquires. With the precision that only a mathematician can muster, Val responds "It's an algorithm that's been. . . iterated."

Then the fun begins. Val goes on to explain that an algorithm is a recipe, that if you knew the recipe to produce a leaf, you could then easily iterate the algorithm to draw a picture of the leaf. "The math isn't difficult. It's what you did at school. You have an x and y equation. Any value for x gives you a value for y. So you put a dot where it's right for both x and y. Then you take the next value for x which gives you another value for y . . . what she's doing is, every time she works out a value for y, she's using *that* as her next value for x. And so on. Like a feedback. . . . If you knew the algorithm, and fed it back say ten thousand times, each time there'd be a dot somewhere on the screen. You'd never know where to expect the next dot. But gradually you'd start to see this shape, because every dot will be inside the shape of this leaf."

2.2 The Chaos Game

What Thomasina has discovered and what Val is trying to explain is what is now commonly called the "chaos game," or, more precisely, an iterated function system. The game proceeds in its simplest formulation as follows. Place three dots at the vertices of a triangle. Color one vertex red, one green, and one blue. Then take a die and color two faces red, two green, and two blue.

To play the game, you need a *seed*, an arbitrary starting point in the plane. The algorithm is: Roll the die, then depending upon which color comes up, move your point half the distance toward the appropriate colored vertex. Then iterate, i.e., repeat this process, using the terminal point of the previous move as the seed for the next. Do not plot the first 15 points generated by this algorithm, but after these few initial moves, begin to record the location of each and every point.

Students who have not seen this game before are always surprised and amazed at the result. Most expect the algorithm to yield a blur of points in the middle of the triangle. Some expect the moving point to fill the whole triangle. But the fact is, the result is anything but a random mess: the resulting picture is one of the most famous of all fractals, the Sierpiński triangle. See Figure 2.1.

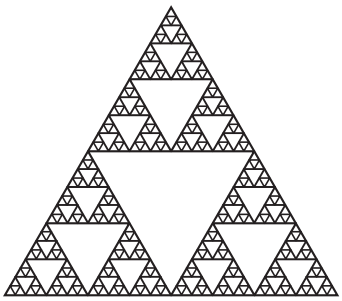


Figure 2.1. The Sierpiński triangle.

The important point about this object being a fractal is that it is a self-similar set. Then, using this self-similarity, we can "go backwards." That is, just by looking at the self-similar features of this set, we can read off the rules of the algorithm that allowed us to generate this set. For the Sierpiński triangle basically consists of three self-similar pieces in which each length is half the length of the corresponding length in the original triangle. And these are precisely the numbers that came up when we generated this image: three vertices, and we moved one-half the distance to each vertex at each iteration.

For leaf-making purposes (which we will describe later), it is best to rework this algorithm in a slightly different form. Begin with a square in the plane, and put the red vertex in the center of the top side of the square, and the other two vertices at the lower vertices. The algorithm then linearly contracts all points in the original square into one of three smaller subsquares by moving half the distance toward the vertex in each subsquare as in Figure 2.2. For example, if the origin in the plane is located at the lower left vertex, then the contraction that takes the square into the region A is simply $(x, y) \mapsto (0.5x, 0.5y)$.

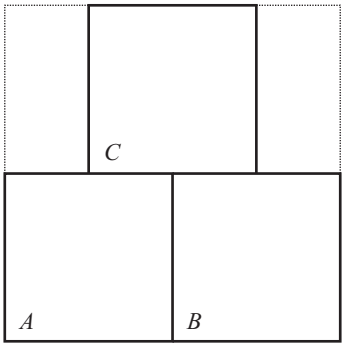


Figure 2.2. The three contractions to produce the Sierpiński triangle. Each move of the chaos game contracts the large outermost square linearly into one of the subsquares.

2.3 Other Chaos Games

As another example of Thomasina's algorithm, start with six points arranged at the vertices of a regular hexagon. Number them from one to six and erase the colors on the die. Beginning with an initial seed in the hexagon, roll the die and then move the starting point two-thirds of the distance toward the appropriate vertex. For what comes later, we think of this as contracting the original distance to the chosen vertex by a factor of three. That is, three is the contraction ratio for this game rather than two as in the case of the Sierpiński triangle.

Then iterate this procedure. As before, do not record the first 15 or so iterations, but plot the rest. The result is again anything but a random mess: It is the Sierpiński hexagon. See Figure 2.3. Note that this fractal image is not quite

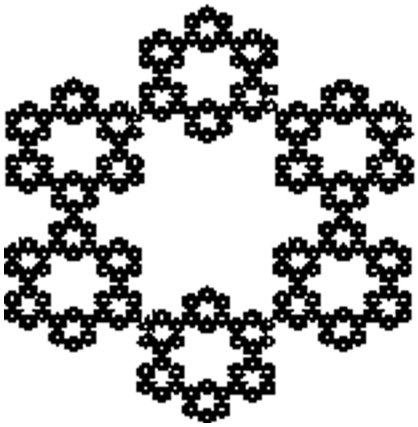


Figure 2.3. The Sierpiński hexagon.

the lifelike image that Thomasina promised us, but there is a hint of what is to come. Look at the boundary of the innermost white region in the Sierpiński hexagon (or, in fact, the boundary of any internal white region). Note how this curve resembles a snowflake. Indeed, this boundary curve is the von Koch snowflake curve, another very famous fractal.

This particular algorithm can also be expressed in terms of linear contractions of a given square. Start with a square and place six points in the square so that they form the boundary of a regular hexagon. Geometry exercise: Where should these points be placed if vertex number one lies in the middle of the left side of the square? At each iteration, contract all points in the original square toward the appropriate vertex. The image is, in each case, a square exactly one-third the size of the original square and located as shown in Figure 2.4.

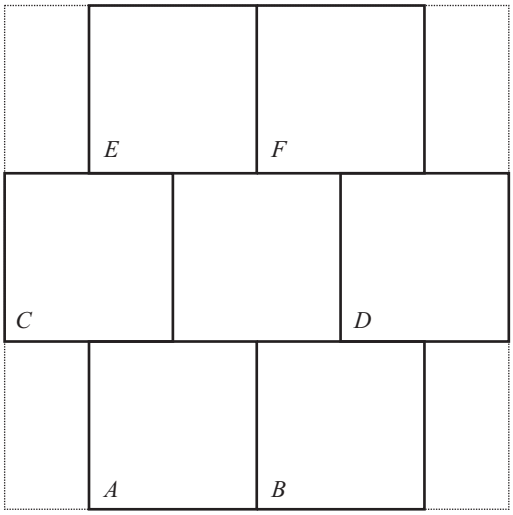


Figure 2.4. The six contractions to produce the Sierpiński hexagon.

And again we can use the self-similar nature of the Sierpiński hexagon to go backwards. For this set consists of six self-similar pieces, each of which is one-third the size of the original hexagon. And those were the numbers that comprised the rules of this game: six vertices and we contracted the distance each time we rolled by one-third.

As another example, suppose we play the chaos game with eight vertices arranged as follows in a square: four lie at the corners of the square and the other four at the midpoints of the sides of the square. With a contraction ratio of three as in the case of the hexagon, we obtain another famous fractal, the Sierpiński carpet depicted in Figure 2.5.

We can complicate things a bit by allowing rotations (and reflections) in the rules of the game. For example, suppose we start with the original configuration that produced the Sierpiński triangle, but we now change one of the rules as

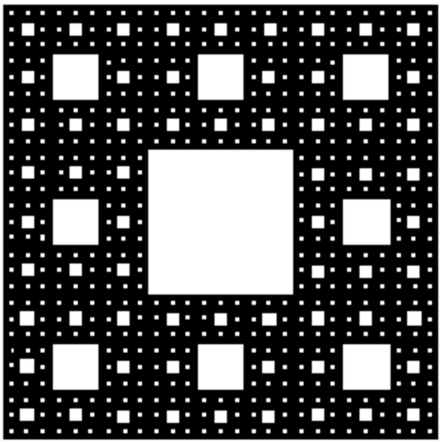
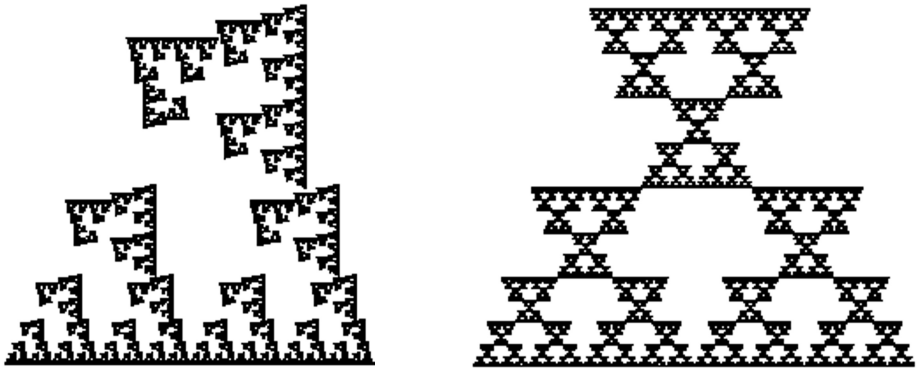


Figure 2.5. The Sierpiński carpet.



A. B.

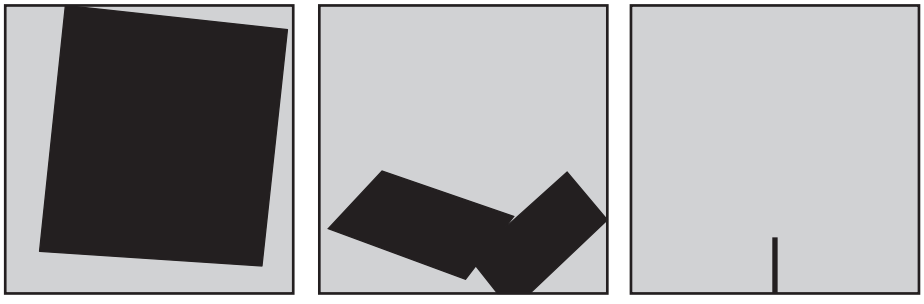
Figure 2.6. The two fractals obtained by adding rotations to the rules.

follows. When we roll the color corresponding to the topmost vertex, we first move the given point half the distance toward that vertex as before, but then we rotate the image point 90 degrees in the counterclockwise direction about the top vertex. (Here we choose a larger square around the chosen vertices than we did originally.) The fractal that emerges is shown in Figure 2.6A. Note again that we can go backwards, since the top self-similar piece of the resulting image is exactly half the size of the entire fractal, but it is now rotated by 90 degrees, while the other two self-similar pieces are also half the size but are not rotated. If we change this rule so that the rotation is 180 degrees about the top vertex, we obtain the image displayed in Figure 2.6B. This fractal consists of three self-similar pieces, each of which is half the size of the entire set, but the top piece is rotated by 180 degrees while the bottom two pieces are not rotated. Again, we can go backwards.

2.4 Thomasina's Geometry

How did Thomasina produce an algorithm that yields a natural form? We will illustrate this with the simplest case, a fern. A leaf is a little more difficult and a little less spectacular than a fern. We need to describe an algorithm that, when iterated as in the chaos game, yields an image of a fern. The fern we will produce is often called the *Barnsley fern* after the mathematician who popularized this procedure [1].

To do this we start with a square as before. We will describe four linear contractions on this square. Unlike the previous two examples, these contractions will involve more than just simple contractions and rotations; they will involve more general linear transformations. Here is the first operation: squeeze and distort the square linearly so that its image appears as in Figure 2.7A. Note that the square is compressed a bit from the bottom and from both sides, and then rotated a little. Figure 2.7B displays the effects of the next two contractions. The left hand parallelogram is obtained by first shrinking the square to a rectangle, then shearing and rotating to the left. The second is obtained in similar fashion, except that the square is first flipped along its vertical axis, and then contracted, sheared, and rotated to the right. In Figure 2.7C we see the final contraction: the entire square is crushed to a line segment in the horizontal direction, then compressed again in the vertical direction to yield the short vertical line segment indicated.



A. B. C.

Figure 2.7. The four contractions that generate the Barnsley fern.

Each of these rules can be described concisely using some matrix algebra. In section 2.6, we give exact formulas for each of these transformations as well as a brief discussion of where they come from.

Now we play the chaos game with these rules as the four constituent moves. However, instead of randomly choosing a particular contraction with equal probabilities, we will choose the rules to apply with differing probabilities. We will apply the first contraction with the highest probability, namely 85% of the time. Contractions 2 and 3 will be invoked with probability .07, and the final contraction will be called with probability only .01. When this algorithm is carried out using a computer, the dots slowly fill the screen to reveal a lifelike image of a fern, as illustrated in Figure 2.8.

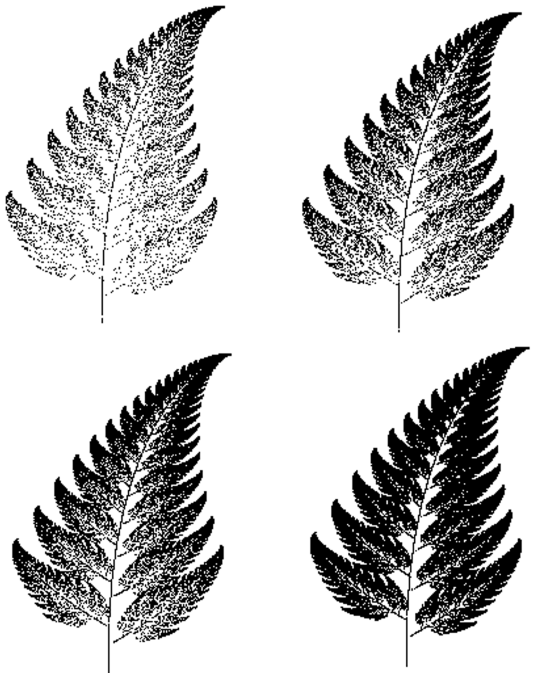


Figure 2.8. The results of the chaos game played with 10,000, 30,000, 50,000, and 150,000 iterations. As Valentine said to Hannah: "If you knew the algorithm, and fed it back say ten thousand times, each time there'd be a dot somewhere on the screen. You'd never know where to expect the next dot. But gradually you'd start to see this shape, because every dot will be inside the shape of this (leaf)."

2.5 Why Thomasina's Algorithm Works

The fact that the above algorithm works is no mystery; it is exactly the same as in the previous examples of chaos games. For we can decompose a fern into "self-similar" copies of itself. Look closely at Figure 2.8: Do you see several pieces of the fern that resemble the entire fern, only in miniature? If you remove the two lowest fronds of the fern and a piece of the stem, then what remains is more or less an exact copy of the original fern, only slightly smaller. Indeed, we can obtain this smaller piece of the fern by taking the entire fern and applying our first contraction to it. That is, we compress the fern from both sides and the bottom and rotate a bit to move it onto the slightly smaller upper piece.

Also, note how the two lowest fronds of the fern are arranged. We may obtain the left-hand frond by compressing the original fern by a much larger amount, and then rotating to the left. The right-hand frond is obtained by first flipping the entire fern along its vertical axis, then contracting and rotating to the right. Finally, the piece of the stem that was removed can be obtained by squashing the entire fern to the center and then down, exactly our contraction number 4.

So we have divided the fern into four "self-similar" copies of itself, i.e., copies that can be obtained by applying the linear rules above. The mathematical theorem behind all of this says that, when we iterate the above rules, the resulting image is exactly what we started with, the fern. We use different probabilities here so that the density of points will be more or less even when the iteration is complete (there are many more points that need to be drawn in the image region given by contraction number 1 than that given by contraction number 4).

2.6 The Formulas

Since the four contractions that produce the fern are affine transformations of the plane, they may be encoded using matrices. Each transformation is of the form

$$V_{\text{new}} = A \cdot V_{\text{old}} + W$$

where V_{old} is the vector representing the seed, V_{new} is the new position of the point, A is a 2×2 matrix, and W is a constant vector. For example, contraction 1 is given by

$$\begin{pmatrix} x_{\text{new}} \\ x_{\text{old}} \end{pmatrix} = \begin{pmatrix} 0.85 & 0.04 \\ -0.04 & 0.85 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} 0 \\ 1.60 \end{pmatrix}.$$

Contractions 2 and 3 take the form

$$\begin{pmatrix} x_{\text{new}} \\ x_{\text{old}} \end{pmatrix} = \begin{pmatrix} 0.20 & -0.26 \\ -0.23 & 0.22 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} 0 \\ 1.60 \end{pmatrix}$$

and

$$\begin{pmatrix} x_{\text{new}} \\ x_{\text{old}} \end{pmatrix} = \begin{pmatrix} -0.15 & 0.28 \\ 0.26 & 0.24 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} 0 \\ 0.44 \end{pmatrix}.$$

Finally, contraction 4 is given by

$$\begin{pmatrix} x_{\text{new}} \\ x_{\text{old}} \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0.16 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{old}} \\ y_{\text{old}} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

The square involved is given by $-5 \leq x \leq 5$ and $0 \leq y \leq 10$.

2.7 Valentine's Grouse

As Thomasina struggles with her new geometry, there is a parallel mathematical development taking place in the play. Valentine is trying to use ideas from chaos theory to explain the rise and fall of the population of grouse on the Sidley Park Estate. He knows the data about grouse kills on the estate for the past two hundred years, and he would like to extrapolate from this to predict the populations in the future. Curiously, he is using the exact same technique that Thomasina had experimented with years before. Well, not quite. As Valentine explains, "Actually I'm doing it from the other end. She started with an equation and turned it into a graph. I've got a graph—real data—and I'm trying to find the equation which would give you the graph if you used it the way she used hers. Iterated it. It's how you look at population changes in biology. Goldfish in a pond, say. This year there are x goldfish. Next year there'll be y goldfish. Some get born, some get eaten by herons, whatever. Nature manipulates the x and turns it into y. Then y goldfish is your starting population for the following year. Just like Thomasina. Your value for y becomes your next value for x. The question is: what is being done to x? What is the manipulation? Whatever it is, it can be written down in mathematics. It's called an algorithm."

One of the simplest such algorithms used by population biologists is the logistic equation given by

$$F_k(x) = kx(1 - x).$$

Here x represents the percentage of some maximal population so that x lies between 0 and 1. The constant k is a parameter; we would use one value of k for grouse, another for rabbits, and a third for elephants. Given an initial population x_0 and a particular value of k, we can then iterate F_k to find the populations in successive years. For

example, if $k = 1.5$ and $x_0 = 0.123$, then we find in succession

$$\begin{aligned} x_0 &= 0.123 \\ x_1 &= 0.161\ldots \\ x_2 &= 0.203\ldots \\ x_3 &= 0.243\ldots\ldots \\ x_4 &= 0.275\ldots \\ &\vdots \\ x_{20} &= 0.3333\ldots \\ x_{21} &= 0.3333\ldots \end{aligned}$$

so that the population has eventually stabilized at $0.3333\ldots$. If, on the other hand, we select $k = 3.2$ and $x_0 = 0.123$, then after several iterations we find that the population begins to cycle back and forth. One year the population is high; the next year it is low:

$$\begin{aligned} x_0 &= 0.123 \\ &\vdots \\ x_{20} &= 0.7994\ldots \\ x_{21} &= 0.5130\ldots \\ x_{22} &= 0.7994\ldots \\ x_{23} &= 0.5130\ldots. \end{aligned}$$

Like Thomasina, we can turn this data into a graph by plotting the *time series* corresponding to this iteration. This is a plot of the iteration count versus the actual numerical values. The cycling behavior above is illustrated in Figure 2.9.

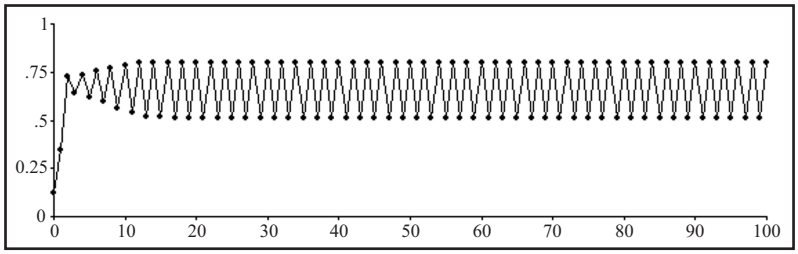


Figure 2.9. A time series indicating cyclic behavior.

When we choose $k = 4$ and $x_0 = 0.123$, the results of iteration are anything but predictable (Figure 2.10). The time series for this iteration shows no pattern whatsoever. More importantly, when we choose a nearby initial population, say 0.124, the output of the iteration is vastly different. A small change in the initial population has produced a major change in the eventual behavior. This is the phenomenon of chaos. Here we see that a very simple iterative scheme can yield results that are totally unpredictable.

2.8 The Orbit Diagram

The logistic equation would seem to be a rather simple mathematical object; after all, it is only a quadratic function. How hard could iteration of a quadratic function be? Well, it's pretty hard! Indeed, mathematicians finally understood the entire picture for the logistic equation in the late 1990's.

The fact is that, when the parameter k is varied, there are a tremendous number of different possibilities that may occur. The orbits of the function may tend to some attracting cycle (as in the case $k = 3.2$ displayed in Figure 2.9). Or the orbits may behave chaotically as in Figure 2.10. When the function is chaotic, lots of different behaviors are

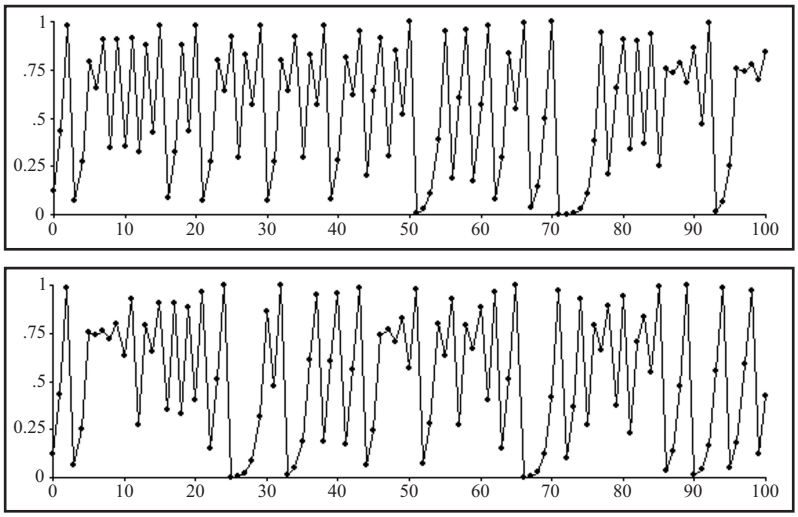


Figure 2.10. Several time series for $k = 4$ indicating chaotic behavior. The first is for the initial value $x_0 = 0.123$, the second for $x_0 = 0.124$.

possible: there are infinitely many orbits that cycle and infinitely many other orbits that do not. In the latter case, some of these orbits may fill up a certain interval (or intervals) densely. And, as in the case $k = 4$, nearby orbits have vastly different fates.

To gain an appreciation of the complexity of the logistic function, we have plotted the *orbit diagram* of this function in Figure 2.11. In this figure, the parameter is plotted on the horizontal axis (here k runs from 2 to 4). Above each k-value, we plot the "eventual" behavior of the orbit of 0.5, the critical point for this function (i.e., the point where the derivative of the logistic function is zero). When we say eventual, we mean that we compute the first 400 points on the orbit of 0.5, but then only plot the last 300 points on this orbit.

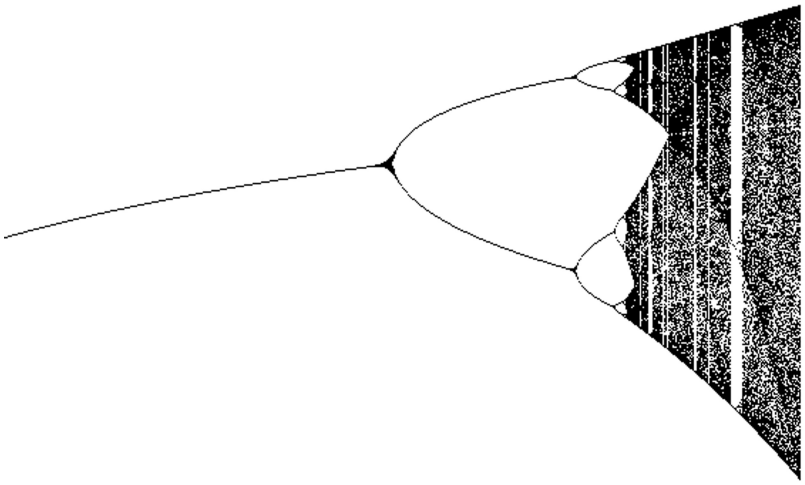


Figure 2.11. The orbit diagram for the logistic function.

As a remark, there is a reason why we use the critical point to plot this picture, for it is a fact that the critical point always finds any "attracting" cycle. Therefore there can be at most one attracting cycle for any chosen k-value; that is, for any specific logistic function, all its other cycles must be "repelling." In the orbit diagram, the "windows" that appear are regions in which the logistic function has an attracting cycle of some given period. There is usually much more going on in these windows. In any window except the largest left-hand window (where we see a fixed point, followed by a 2-cycle, then a 4-cycle, etc.), there is in fact an uncountable set of points on which the function behaves chaotically. For example, a famous theorem due to Sharkovsky states that, when a continuous function on the real line has a cycle of period three, then it must have a cycle of every other period as well! So, in the right of the orbit diagram,

we see a period three window. For any parameter drawn from this region, the logistic map must therefore have cycles of all periods as well as a set on which there is chaotic behavior.

Curiously, the way mathematicians finally understood the logistic function was by moving to iteration in the complex plane. There we have many more tools available (the Schwarz Lemma, the Riemann Mapping Theorem, etc.). Using these tools together with computer graphics enabled mathematicians to describe precisely the order of events that occurred as the parameter k-varies. Indeed, it is the Mandelbrot set that provides this explanation (technically, the Mandelbrot set is a record of all that occurs for another quadratic function, namely $z^2 + c$, but this function and the logistic function behave essentially the same from a dynamical systems point of view). And, of course, the Mandelbrot set (see Figure 2.12) makes a brief appearance in the play. For Hannah glances over Valentine's shoulder and catches a glimpse of what is on the computer screen. "Oh, but… How beautiful!" she exclaims. Val responds, "The Coverly set. Lend me a finger. (He takes her finger and presses one of the computer keys several times.) See? In an ocean of ashes, islands of order. Patterns making themselves out of nothing. I can't show you how deep it goes. Each picture is a detail of the previous one, blown up. And so on. Forever. Pretty nice, eh?" Hannah: "Is it important?" Val responds, "Interesting. Publishable." "Well done!" says Hannah. "Not me. It's Thomasina's. I just pushed her equations through the computer a few million times further than she managed to do with her pencil."

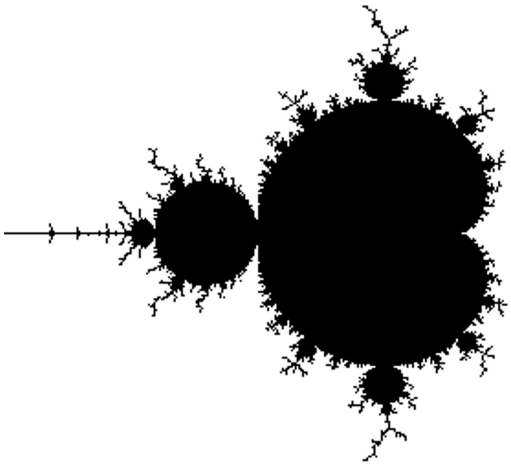


Figure 2.12. The Mandelbrot set.

2.9 Summary

All of the different mathematical ideas in the play are tremendously exciting, since they have, for the most part, arisen in the past quarter century. Moreover, the images with which they are associated are extremely beautiful and captivating. Valentine himself is ecstatic about this stuff. He summarizes what he is seeing in chaos and fractals: "The unpredictable and the predetermined unfold together to make everything the way it is. It's how nature creates itself, on every scale, the snowflake and the snowstorm. It makes me so happy. To be at the beginning again, knowing almost nothing... A door like this has cracked open five or six times since we got up on our hind legs. It's the best possible time to be alive, when almost everything you thought you knew is wrong."

The play *Arcadia* is a wonderful experience for all students. It is fast-paced, witty, and thoroughly enjoyable. Best of all, it can be combined with some wonderful mathematical ideas to give students a truly interdisciplinary experience. I have worked over the years with several high school, college, and professional productions of *Arcadia*. Often, in the high schools, the mathematics, science, and humanities teachers team up to give introductory classes on the topics of the play to many or all of the students in the school. These various interdisciplinary experiences for the students are then brought to a wonderful conclusion with the staging of the play.

For more information on the history of the subject of chaos, consult [6]. Some of the mathematical and biological underpinnings of the subject may be found in [7]. Fractals are described in [1], [3], and [8]. A primer on chaos can be found in [2] for high school students and in [4] for undergraduates. An interactive version of this paper plus some

java applets to play the chaos game may be found at the Dynamical Systems and Technology Project website at math.bu.edu/DYSYS.

Bibliography

[1] Barnsley, M., *Fractals Everywhere*, Boston: Academic Press, 1988.

[2] Choate, J. and R. L. Devaney, *Chaos*, Emeryville, CA: Key Curriculum Press, 1999.

[3] Choate, J., R. L. Devaney, and A. Foster, *Fractals*, Emeryville, CA: Key Curriculum Press, 1999.

[4] Devaney, R. L., *A First Course in Chaotic Dynamical Systems*, Westview Press, 1992.

[5] ——, "Chaos Rules!" *Math Horizons*, XII(2), (November 2004), 11–14.

[6] Gleick, J., *Chaos: Making a New Science*, New York: Viking, 1987.

[7] May, R. B., "Simple mathematical models with complicated dynamics," *Nature*, 261(1976), 459–467.

[8] Mandelbrot, B., *The Fractal Geometry of Nature*, San Francisco: Freeman, 1983.

[9] Stoppard, T., *Arcadia*, London: Faber and Faber, 1993.

3

Excursions Through a Forest of Golden Fractal Trees

T. D. Taylor
St. Francis Xavier University

Abstract. This paper presents an exploration of various features of the four self-contacting symmetric binary fractal trees that scale according to the golden ratio. This excursion demonstrates some beautiful connections in math by providing a wealth of interesting exercises involving geometry, trigonometry, infinite series, the Fibonacci sequence, self-similarity and symmetry.

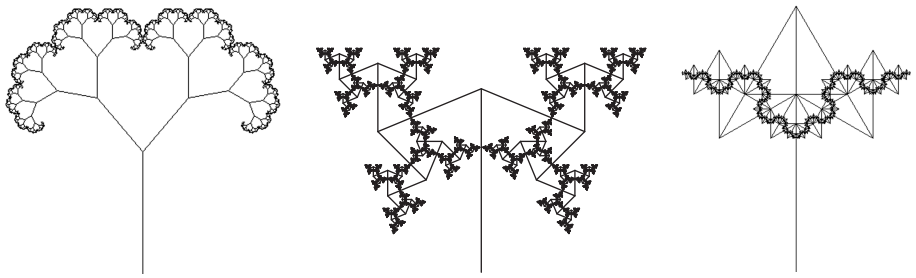
3.1 Introduction

This paper presents an exploration of various features of the four self-contacting symmetric binary trees that scale according to the golden ratio. We begin with an introduction to background material. This includes relevant definitions, notations and results regarding symmetric binary fractal trees; various aspects of the golden ratio; and connections between fractals and the golden ratio. The main part of the paper consists of four subsections, with each subsection discussing a particular 'golden tree'. Each tree possesses remarkable symmetries. Classical geometrical objects such as the pentagon and decagon make appearances, as do fractal objects such as a golden Cantor set and a golden Koch curve. We find new representations of well-known facts about the golden ratio by using the scaling nature of the trees.

3.1.1 Symmetric Binary Fractal Trees

Fractal trees were first introduced by Mandelbrot in "The Fractal Geometry of Nature" [8]. In general, fractal trees are compact connected subsets of $\mathbb{R}^n$ (for some $n \geq 2$) that exhibit some kind of branching pattern at arbitrary levels. The class of symmetric binary fractal trees was more recently studied by Mandelbrot and Frame [9]. A symmetric binary fractal tree $T(r, \theta)$ is defined by two parameters, the scaling ratio r (a real number between 0 and 1) and the branching angle θ (an angle between 0° and 180°). The trunk splits into two branches, one on the left and one on the right. Both branches have length equal to r times the length of the trunk and form an angle of θ with the affine hull of the trunk. (The affine hull of a line segment is the line that is collinear with the line segment, and it is sometimes referred to as the linear extension.) Each of these branches splits into two more branches following the same rule, and the branching is continued *ad infinitum* to obtain the fractal tree. Each tree has left-right symmetry, with the affine hull as the line of reflection.

A self-contacting symmetric binary fractal tree has self-intersection but no actual branch crossings. For a given branching angle θ, there is a unique scaling ratio $r_{sc}(\theta)$, or just r_{sc}, such that the corresponding tree is self-contacting [9]. The values of r_{sc} as a function of θ have been completely determined in previous literature [9]. For the purposes of the current paper it is worth noting that the value of r_{sc} is determined as follows: find the smallest scaling ratio such that there is part of the tree besides the trunk that is on the affine hull of the trunk [9], [12]. We shall see how to do this for four specific angles. Figure 3.1 displays three different self-contacting symmetric binary fractal trees. Further details about the general class of self-contacting trees are available [9], [11], [12].



(a) $T(r_{sc}, 40°)$ (b) $T(r_{sc}, 112.5°)$ (c) $T(r_{sc}, 150°)$

Figure 3.1. Self-contacting Symmetric Binary Fractal Trees

The remainder of this subsection is devoted to introducing terminology and notations for various features of the fractal trees. This will make it easier to discuss the specific trees in Section 3.2. The reader is encouraged to refer back to this subsection.

To describe the fractal trees and their scaling properties more precisely, we use an address system. For any $r \in (0, 1)$ and any $\theta \in (0°, 180°)$, the *generator maps* m_R and m_L are:

$$m_R\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = r \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.1}$$

$$m_L\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = r \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.2}$$

The map m_R acts on a compact subset of $\mathbb{R}^2$ by shrinking it by a factor of r, then rotating it θ clockwise, and translating it up 1 unit. The map m_L has the same action except that the rotation is counter-clockwise.

An *address* $\mathbf{A} = A_1 A_2 \cdots$ is a string (finite or infinite) of elements, with each element either R (for 'right') or L (for 'left'). The *level* of the address is equal to the number of elements in the string. The only level 0 address is the empty address $\mathbf{A_0}$. An address map $m_{\mathbf{A}}$ is a composition (finite or infinite) of generator maps. An infinite address map is the limit of finite address maps, and is well defined because $r \in (0, 1)$. If C is any compact subset of $\mathbb{R}^2$ and if $m_{\mathbf{A}}$ is a level k address map, then the set $m_{\mathbf{A}}(C)$ is a compact subset of $\mathbb{R}^2$ that is similar to C with contraction factor r^k. This fact will be useful for defining certain parts of a tree, such as branches and subtrees.

The *trunk* T_0 of any tree is the closed vertical line segment between the points $(0, 0)$ and $(0, 1)$. The image $m_R(T_0)$ is the first branch to the right, *i.e.,* the line segment between the point $(0, 1)$ and $(r \sin\theta, 1 + r\cos\theta)$ and the image $m_L(T_0)$ is the first branch on the left side. Given an address $\mathbf{A}$, $m_{\mathbf{A}}(T_0)$ is a branch. The *point with address* $\mathbf{A}$, $P_{\mathbf{A}}$, is $m_{\mathbf{A}}((0, 1))$. If $\mathbf{A}$ is finite, the point is a *vertex*; if it is infinite, the point is a *tip point*. The *symmetric binary fractal tree* $T(r, \theta)$ is defined to be the limit as the number of levels of branching goes to infinity. The *top points* of a tree are all points of the tree that have maximal y-value. For an address $\mathbf{A} = A_1 A_2 \cdots$, there exists a natural path on the tree that starts with the trunk and goes to $P_{\mathbf{A}}$, consisting of the trunk along with all branches $b(\mathbf{A}_i)$, where $\mathbf{A}_i = A_1 \cdots A_i$. We denote this path $p(\mathbf{A})$. A *level k subtree* of a tree $T(r, \theta)$ is $m_{\mathbf{A}}(T)$ for some level k address $\mathbf{A}$, denoted by $S_{\mathbf{A}}(r, \theta)$ or $S_{\mathbf{A}}$. Each tree is equal to its reflection across the y-axis. Because of this left-right symmetry, we often restrict our attention to the right side of the tree.

3.1.2 The Golden Ratio

The 'golden ratio' ϕ is a remarkable number that arises in various areas of mathematics, nature and art [7], [4], [13], [3], [5], [2].

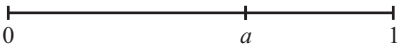


Figure 3.2. Dividing the unit interval according to the golden ratio

The most basic geometric description of the golden ratio involves a line segment. Without loss of generality, assume the line segment is $[0, 1]$. There exists a number $a \in (0, 1)$ such that the ratio of the length of $[0, 1]$ to the length of $[0, a]$ is equal to the ratio of $[0, a]$ to $[a, 1]$ (see Figure 3.2). This ratio is denoted by ϕ. Setting the ratios equal gives

$$\phi = \frac{1}{a} = \frac{a}{1-a}.$$

Thus ϕ is the unique positive solution to the quadratic equation $x^2 - x - 1 = 0$ where $x = \frac{1}{a}$. Hence

$$\phi = \frac{1+\sqrt{5}}{2} \approx 1.618034... \tag{3.3}$$

Immediately we have the following:

$$\frac{1}{\phi} = \frac{-1+\sqrt{5}}{2} = \phi - 1 \approx 0.618034.... \tag{3.4}$$

In general, any line segment that is split into 2 pieces of unequal length such that the ratio of the length of the whole segment over the length of the longer piece is equal to the ratio of the length of the longer piece over the length of the shorter piece, is said to be divided according to the golden ratio.

Another geometric description is given by the golden rectangle, as displayed in Figure 3.3. Given a rectangle having sides in the ratio $1 : \phi$, ϕ is such that partitioning the original rectangle into a square and a new rectangle results in the new rectangle having sides with a ratio $1 : \phi$. That is, the original rectangle and the new smaller rectangle have the same relative proportions.

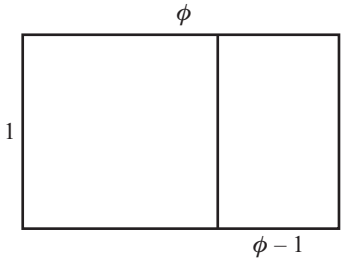


Figure 3.3. The Golden Rectangle

Before discussing other geometric aspects of the golden ratio, we will look at some algebraic properties. There are many interesting and useful equations that can be derived. Because $\phi + 1 = \phi^2$, any positive power of ϕ can be expressed in terms of a linear expression with ϕ. For example:

$$\phi^3 = \phi\phi^2 = \phi(1+\phi) = \phi + \phi^2 = \phi + \phi + 1 = 2\phi + 1 \tag{3.5}$$

Similarly, $\phi^4 = 3\phi + 2$.

Recall the Fibonacci sequence:

$$1, 1, 2, 3, 5, 8, \cdots \tag{3.6}$$

The Fibonacci numbers F_n, $n \geq 0$, are recursively defined by

$$F_0 = 1, \quad F_1 = 1, \quad F_n = F_{n-1} + F_{n-2} \ (n \geq 2). \tag{3.7}$$

It is a nice induction exercise to show that

$$\phi^n = F_{n-1}\phi + F_{n-2} \ \ (n \geq 2). \tag{3.8}$$

For reciprocal powers of ϕ, we can also find linear equations by using the fact that $1/\phi = \phi - 1$. Then we have

$$\frac{1}{\phi^n} = (-1)^n(F_n - F_{n-1}\phi) \ \ (n \geq 1). \tag{3.9}$$

Now consider the successive ratios of the Fibonacci numbers:

$$\begin{array}{ll} \frac{F_1}{F_0} = \frac{1}{1} = 1, & \frac{F_2}{F_1} = \frac{2}{1} = 2 \\ \frac{F_3}{F_2} = \frac{3}{2} = 1.5, & \frac{F_4}{F_3} = \frac{5}{3} = 1.\overline{6} \\ \frac{F_5}{F_4} = \frac{8}{5} = 1.6, & \frac{F_6}{F_5} = \frac{13}{8} = 1.625 \\ \frac{F_7}{F_6} = \frac{21}{13} \approx 1.615, & \frac{F_8}{F_7} = \frac{34}{21} \approx 1.619 \end{array} \tag{3.10}$$

It seems like the successive ratios have a limit, and indeed it is a nice exercise using the recursive definition of F_n to show that

$$\phi = \lim_{n\to\infty} \frac{F_n}{F_{n-1}}.$$

There are many other interesting aspects of the golden ratio ϕ. The golden ratio ϕ can be considered to be the 'most irrational' number because it has a continued fraction representation

$$\phi = [1, 1, 1, \cdots] = 1 + \frac{1}{1 + \frac{1}{1+\frac{1}{1+\cdots}}}. \tag{3.11}$$

This representation is straightforward to derive by using the equation $\phi = 1 + 1/\phi$.

There is also a nested radical expression for the golden ratio which can be derived from the equation $\phi^2 = \phi + 1$. Define a sequence $\{s_n\}$ by

$$s_1 = 1, \ \ s_n = \sqrt{1 + s_{n-1}}, \ \ n \geq 2$$

This sequence is monotone increasing and bounded above (two other nice exercises using induction) and is clearly bounded below. Thus, by the Monotone Convergence Theorem, the sequence is convergent, and one can show that the limit is indeed ϕ:

$$\phi = \sqrt{1 + \sqrt{1 + \sqrt{1 + \cdots}}}. \tag{3.12}$$

Returning to geometry, we find the golden ratio in many classical geometrical figures. We briefly mention figures that are relevant for this paper. They will all reappear when we study the golden fractal trees.

First consider an equilateral triangle and the circle that circumscribes it, as in Figure 3.4. If the line segment AB joining the midpoints of two sides is extended to the circle at C, then the ratio of AC to AB is ϕ.

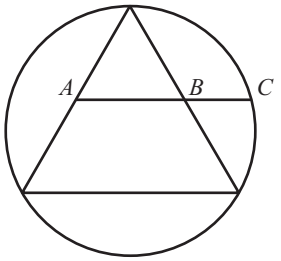


Figure 3.4. Equilateral triangle

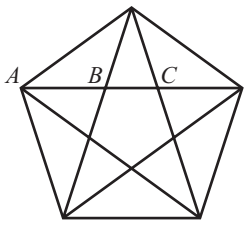


Figure 3.5. Pentagon and Pentagram

Now consider a pentagon and a pentagram with the same vertices, as in Figure 3.5. The interior angles at each vertex of a pentagon are 108°. It is a nice geometry exercise to show that the ratio of AC to AB is ϕ.

A triangle with the angles $72°, 72°, 36°$ is commonly called a *golden triangle* ($\triangle ABC$ in Figure 3.6a). Such a triangle can be divided into a smaller golden triangle ($\triangle BCD$) and a *golden gnomon* (a triangle with the angles $36°, 36°, 108°$, given by $\triangle ABD$). Any triangle in Figure 3.5 is either a golden triangle or a golden gnomon. Golden triangles also appear in the decagon (Figure 3.6b).

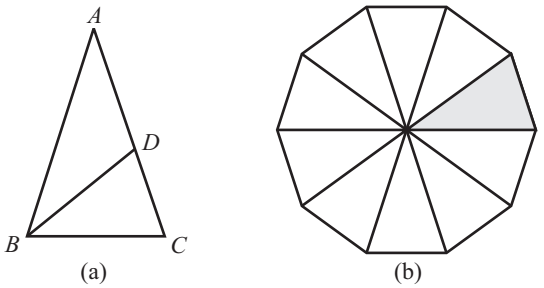


Figure 3.6. (a) Golden Triangles ($\triangle ABC$ and $\triangle BCD$) and Golden Gnomon ($\triangle ABD$), (b) Decagon

3.1.3 Fractals and The Golden Ratio

The golden ratio appears in fractal geometry as well as classical geometry, perhaps due to its own iterative (repeating) nature. An iterative nature is apparent in a pentagon and pentagram with the same vertices (as in Figure 3.5). There is a smaller pentagon inside, and one can then construct the smaller pentagram with the same vertices as this smaller pentagon. This process can be repeated *ad infinitum*. The same idea applies to a golden triangle as one divides it into a smaller golden triangle and golden gnomon. Besides geometric representations of iterative processes, the iterative nature of the golden ratio is also displayed in the continued fraction representation (Equation 3.11) and the nested radical (Equation 3.12).

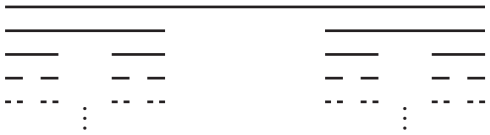


Figure 3.7. Middle Thirds Cantor Set

One well-known fractal is the middle thirds Cantor set, where one starts with an interval of unit length, removes the middle third, and continues the process *ad infinitum* (Figure 3.7). This set is *self-similar* (the union of scaled down versions of itself with no overlaps or gaps), and one can find the corresponding similarity dimension D as follows. From each row to the next, a line segment is scaled down by a factor of 3 to obtain 2 similar line segments, so D is given by $3^D = 2$, or $D = \log 2/\log 3 \approx 0.6309$. We shall see that a 'golden' Cantor set [6] appears in several of our golden trees. The general family of middle Cantor sets $\{C_\alpha\}_{\alpha\in(0,1)}$ includes the classic middle thirds Cantor set. Given $\alpha \in (0, 1)$, the middle-α Cantor set C_α is obtained from removing the middle open set of length α from the unit interval, then continuing to remove middles *ad infinitum*. It is straightforward to show that the similarity dimension of a middle-α Cantor set is $\log 2/\log(1/\beta)$, where $\beta = (1-\alpha)/2$. There are so many other instances of the golden ratio and fractals appearing together in literature that it is not possible to provide a complete survey here. There is an interesting relationship between the golden ratio and variations on another well-known fractal, the Sierpiński gasket [1]. Our current paper deals with the golden ratio and self-contacting symmetric binary fractal trees.

3.2 The Golden Trees

We now discuss the relationship between the golden ratio and self-contacting symmetric binary fractal trees. There are exactly four self-contacting symmetric binary fractal trees for which $r_{sc} = 1/\phi$ [11]. The four branching angles are 60°, 108°, 120° and 144°. We shall discuss each tree separately, though there are some common features. Each tree possesses remarkable symmetries in addition to the usual left-right symmetry of the symmetric binary trees. Classical geometrical objects such as the pentagon and decagon, which are related to the golden ratio as discussed above, make an appearance. We shall also see that special subsets of the trees are other fractals related to the golden ratio, such as a golden Cantor set and a golden Koch curve. Using the scaling nature of the trees, we find new representations of well-known facts about the golden ratio. The trees all seem to 'line up'. Some of the observations are given without proof, and generally the proofs are wonderful exercises involving trigonometry, geometric series, the scaling nature of fractal trees, and the many special equations involving the golden ratio.

3.2.1 Golden 60

The first golden tree that we present is $T(r_{sc}, 60°)$. This particular tree has been discussed in other literature [13], [10]. We begin by demonstrating that $r_{sc} = 1/\phi$.

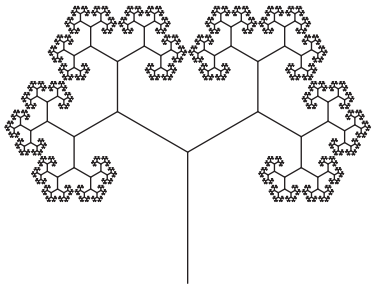


Figure 3.8. $T(1/\phi, 60°)$

To determine r_{sc}, we find a path which leads to a point on the subtree S_R that has minimal distance to the y-axis (other than the point that is the top of the trunk). Then we find the scaling ratio that places this point on the y-axis.

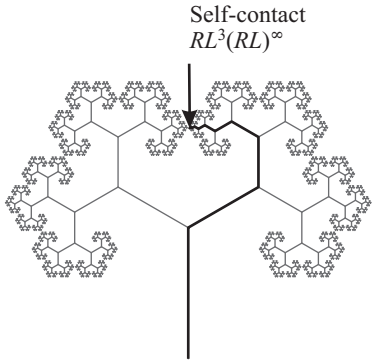


Figure 3.9. Path $p(RL^3(RL)^\infty)$ to self-contact on $T(1/\phi, 60°)$

Recall from the definition of symmetric binary fractal trees that the bottom of the trunk is on the x-axis and the trunk is on the y-axis. The path to such a point is given by $p(RL^3(RL)^\infty)$; this path is displayed in Figure 3.9. Recall that $P_{\mathbf{A}}$ denotes the point with address **A**, and let $P_c = (x_c, y_c)$ denote $P_{RL^3(RL)^\infty}$. We use a 'c' to denote 'contact'. Then to find the value of the x-coordinate, we add up the horizontal components of the branches along the path $p(RL^3(RL)^\infty)$.

$$\begin{aligned} x_c &= r\sin\theta - r^3\sin\theta - r^4\sin(2\theta) - r^5\sin\theta - r^6\sin(2\theta)\cdots \\ &= r\sin\theta - r^3(\sin\theta + r\sin(2\theta))(1 + r^2 + \cdots) \\ &= r\sin\theta - \frac{r^3}{1-r^2}(\sin\theta + 2r\sin\theta\cos\theta) \\ &= \frac{r\sin\theta}{1-r^2}[(1-r^2) - r^2(1 + 2r\cos\theta)] \end{aligned}$$

Substituting the value of $\theta = 60°$:

$$\begin{aligned} x_c &= \frac{\sqrt{3}r}{2(1-r^2)}[1 - 2r^2 - r^3] \\ &= \frac{\sqrt{3}r}{2(1-r^2)}(1+r)(1-r-r^2) \end{aligned}$$

The value of r_{sc} that gives the solution to $x_c = 0$ is the root of $1 - r - r^2$ in $(0, 1)$. As mentioned in Subsection 3.1.2, the unique positive root of this quadratic is $r_{sc} = 1/\phi$.

General Note. For any self-contacting tree, a *self-contact point* refers to a point on the tree that is on the y-axis that corresponds to more than one address. For the tree $T(1/\phi, 60°)$, a self-contact point is $P_{RL^3(RL)^\infty}$, because this point is the same as $P_{LR^3(LR)^\infty}$. A self-contact point of a subtree is on the affine hull of the trunk of the subtree and corresponds to more than one address. The two corner points of a tree are $P_{(RL)^\infty}$ and $P_{(LR)^\infty}$, and corner points of a subtree $S_{\mathbf{A}}$ are just the images of these two points under the mapping $m_{\mathbf{A}}$, *i.e.*, the points $P_{\mathbf{A}(RL)^\infty}$ and $P_{\mathbf{A}(LR)^\infty}$.

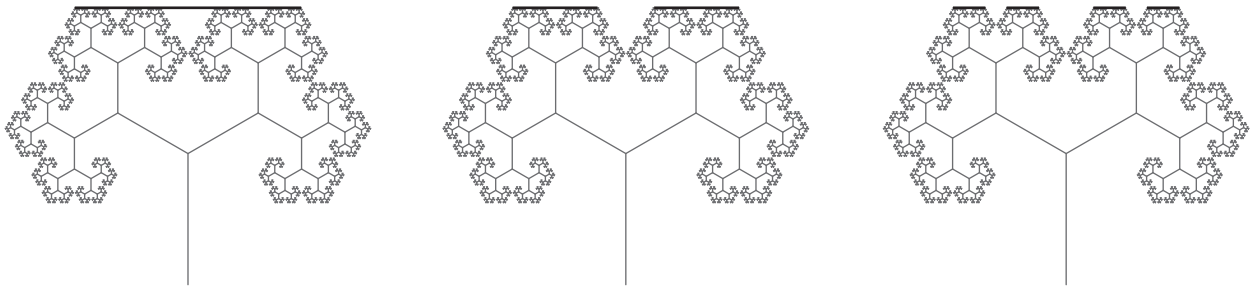


Figure 3.10. Creation of the Golden Cantor Set on $T(1/\phi, 60°)$

The top points of this tree form a middle Cantor set, as is the case for all other self-contacting trees with branching angle less than 135° [12]. Let l be the length of the top of the tree, *i.e.*, the length between the leftmost top point $P_{(LR)^\infty}$ and the rightmost top point $P_{(RL)^\infty}$. It can easily be shown that $l = \sqrt{3}$. The top points are geometrically similar (with a factor of $\sqrt{3}$) to the middle Cantor set with $\alpha = 1/\phi^3$. The first two iterations of removing open middles are shown in Figure 3.10. At the first iteration, the two remaining line segments each have length $l(1/\phi^2) = l/\phi^2$, because they are on the tops of level 2 subtrees S_{RL} and S_{LR}. The length of the gap at the first iteration must be equal to $l(1/\phi^3) = l/\phi^3$. This is because the gap side is the third side of an equilateral triangle with the other two sides being the top of level 3 subtrees S_{RLL} and S_{LRR}. Thus, after simplifying, this expression is equivalent to

$$2\phi + 1 = \phi^3,$$

as we established in Equation 3.5. Thus we have a visual representation on the self-contacting fractal tree of one of the many equations involving ϕ. The scaling dimension of this middle Cantor set is $\log 2/(2\log\phi)$.

The curve on the tree between the rightmost top point and and the leftmost top point can be considered a 'golden' Koch curve (see Figure 3.11). In the first iteration, the original line segment of length $l = \sqrt{3}$ between the points $P_{(LR)^\infty}$ and $P_{(RL)^\infty}$ is replaced with four line segments. Two of these segments have length l/ϕ^2 (because they are the tops of level 2 subtrees) and the other two have length l/ϕ^3 (because they are the tops of level 3 subtrees).

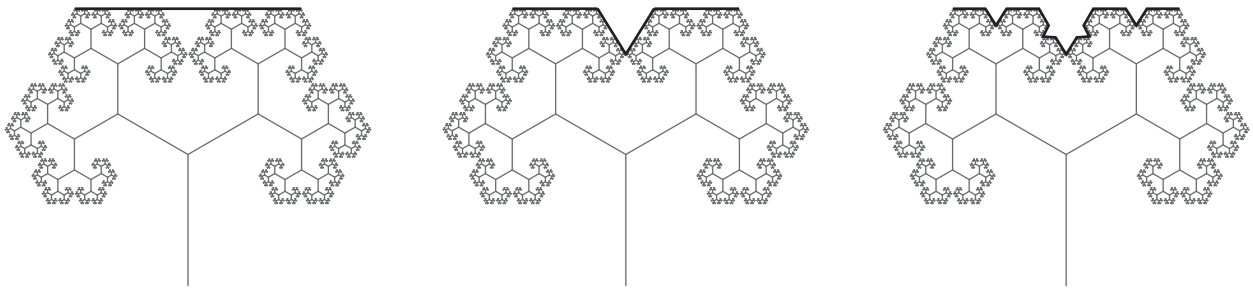


Figure 3.11. Creation of the Golden Koch Curve on $T(1/\phi, 60°)$

Another well-known equation involving ϕ can be visualized on $T(1/\phi, 60°)$. Because $\theta = 60°$, the tops of the subtrees S_{LRR} and S_{RLLL} are collinear. Consider the triangle consisting of the top of the subtree S_{LR} as one side, the union of the tops of S_{LRR} and S_{RLLL} as another side, and the line segment that completes the triangle, as in Figure 3.12. This triangle is an isosceles triangle because the angles are $120°, 30°, 30°$. One side of the triangle has length l/ϕ^2 and the other side has length equal to $l/\phi^3 + l/\phi^4$ because it consists of the top of the level 3 subtree S_{LRR} together with the top of the level 4 subtree S_{RLLL}. Hence

$$\frac{l}{\phi^2} = \frac{l}{\phi^3} + \frac{l}{\phi^4},$$

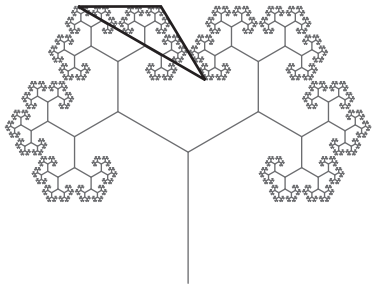
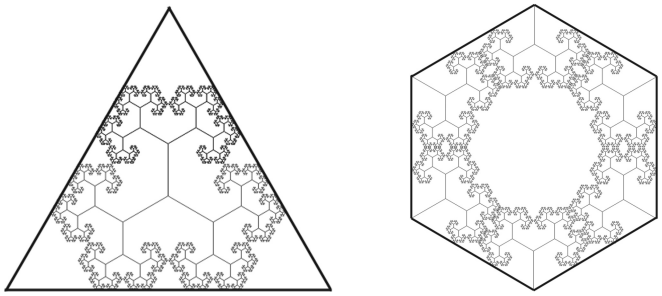


Figure 3.12. Isosceles triangle on $T(1/\phi, 60°)$ demonstrating $\phi^2 = \phi + 1$

which is equivalent to the familiar equations

$$1 = \frac{1}{\phi} + \frac{1}{\phi^2} \text{ or } \phi^2 = \phi + 1.$$

Not surprisingly, there is a connection between this golden tree with branching angle 60 ° and equilateral triangles. Figure 3.13a displays the tree together with two copies of itself, one copy rotated 120 ° to the right around the origin and the other copy rotated 120° to the left around the origin. The trees contact each other at tip points but there are no branch crossings. This triple tree object is contained in an equilateral triangle with sides of length $l(2 + \phi)$, where l is the length of the top of the tree.

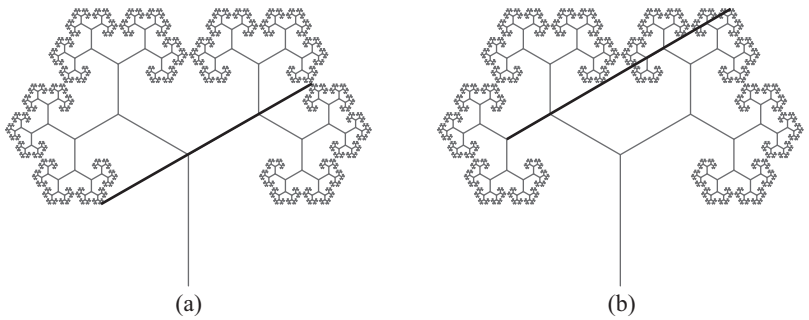


(a) (b)

Figure 3.13. (a) Equilateral triangle and $T(1/\phi, 60°)$ and (b) Hexagon tiled with copies of $T(1/\phi, 60°)$

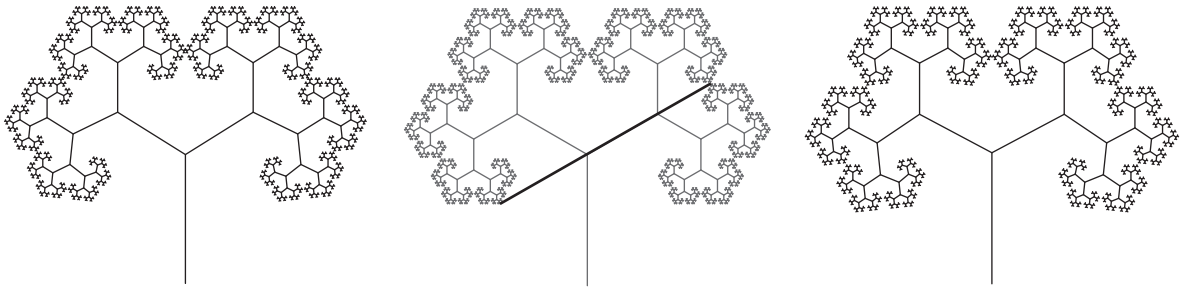
Another regular polygon that naturally relates to $T(1/\phi, 60°)$ is the hexagon, as in Figure 3.13b. In this figure, each vertex is the bottom of the trunk of a tree. The trees intersect, but there are no branch crossings. The boundary of the center region is a golden Koch snowflake (comprised of golden Koch curves as described above).

The golden trees are also special in the sense that branches seem to line up together. In the case of $T(1/\phi, 60°)$, consider Figure 3.14. The affine hull of the branch $b(R)$ (shown in Figure 3.14a) intersects the tree at the contact point of the subtree S_R and at the top left corner point of the subtree S_{LLL} (besides the branch itself). The affine hull of the branch $b(LL)$ (shown in Figure 3.14b) is also the affine hull of the branch $b(RLLL)$ and any other branch of the form $b(RL)^k LL$, for $k \geq 0$. The proof of these claims is a nice exercise involving trigonometry, geometric series and properties of the golden ratio.



(a) (b)

Figure 3.14. Lining up of branches in $T(1/\phi, 60°)$



(a) $T(r_{sc}, 58°)$ (b) $T(1/\phi, 60°)$ (c) $T(r_{sc}, 62°)$

Figure 3.15. Comparison of $T(1/\phi, 60°)$ and two other self-contacting trees with branching angles close to 60 °

For the sake of comparison, Figure 3.15 displays $T(1/\phi, 60°)$ along with the self-contacting trees with branching angles 58° and 62°. The dark line in Figure 3.15(b) is at an angle of 60° with the horizontal.

3.2.2 Golden 108

The next golden tree we look at is the self-contacting tree with branching angle 108 °. As mentioned above in Subsection 3.1.2, the interior angles at each vertex of a pentagon are 108°. Thus it seems natural that pentagons, and in turn the golden ratio, are relevant for the self-contacting tree with this angle.

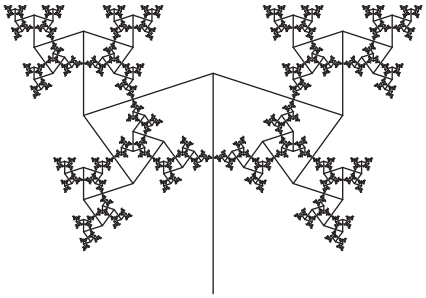


Figure 3.16. $T(1/\phi, 108°)$

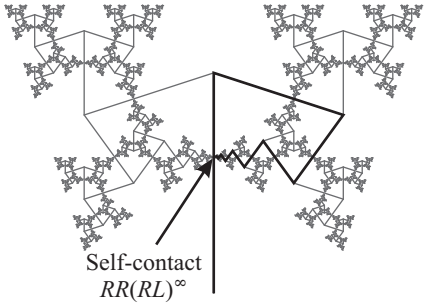
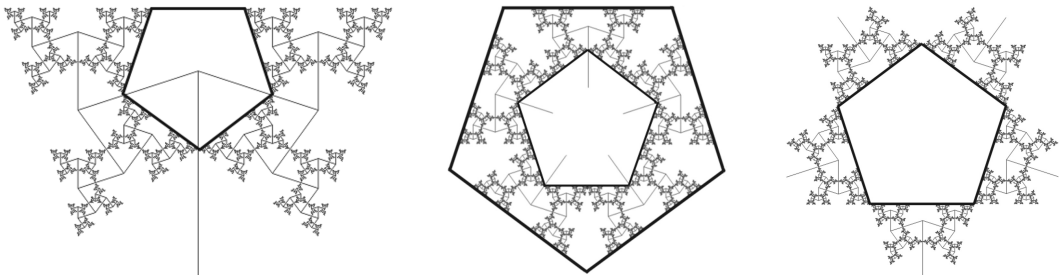


Figure 3.17. Path $p(RR(RL)^\infty)$ to self-contact on $T(1/\phi, 108°)$

The path to a point on the subtree S_R with minimal distance to the y-axis is $p(RR(RL)^\infty)$, as displayed in Figure 3.17. Let $P_c = (x_c, y_c)$ denote $P_{RR(RL)^\infty}$. It is left to the reader to show that the value of $r \in (0, 1)$ that gives the solution to $x_c = 0$ is $r_{sc} = 1/\phi$.

As with $T(1/\phi, 60°)$, the top points of $T(1/\phi, 108°)$ form a middle Cantor set with scaling dimension $\log 2/(2 \log \phi)$. Let l be the length of the top of the tree, *i.e.*, the distance between $P_{(LR)^\infty}$ and $P_{(RL)^\infty}$. It can be shown that $l = 2 \sin 108° = \sqrt{4\phi^2 - 1}/(2\phi) \approx 1.902$. The gap between $P_{LR(RL)^\infty}$ and $P_{RL(LR)^\infty}$ has length l/ϕ^3. Moreover, one can form a pentagon from the tops of the level 3 subtrees S_{LRR}, S_{LLL}, S_{RRR} and S_{RLL} along with the line segment in the gap (as shown in Figure 3.18a).



(a) (b) (c)

Figure 3.18. (a) Pentagon formed from the tops of level 3 subtrees in $T(1/\phi, 108°)$, (b) Outer and inner pentagons around $T(1/\phi, 108°)$ and four copies, (c) Another pentagon from five copies of $T(1/\phi, 108°)$

There is another relationship between this tree and the pentagon. Consider the tree along with four copies of itself, as shown in Figure 3.18b. As with the tiling of an equilateral triangle with copies of $T(1/\phi, 60\,°)$, we have a tiling of the pentagon. The trees contact each other but there are no branch crossings. Finally, Figure 3.18c displays another pentagon that can be associated with five copies of the tree.

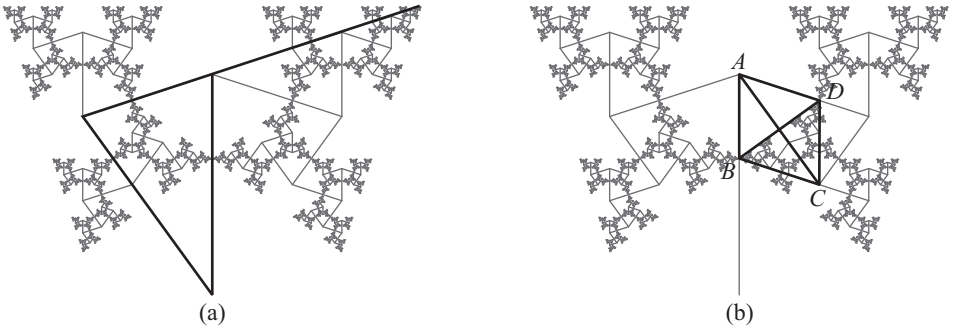


Figure 3.19. (a) Golden triangle and lining up of branches in $T(1/\phi, 108\,°)$, (b) Golden gnomons of $T(1/\phi, 108\,°)$

Besides the pentagon, there is another geometrical figure associated with the golden ratio that is related to this tree. This object is the golden triangle, as shown in Figure 3.19a. The sides of the triangle in the figure consist of the trunk, the branch $b(L)$, and the extension of the branch $b(LL)$ to the trunk. Two of the angles are $72°$, and thus the triangle is isosceles. This same figure also displays the fact that certain branches share the same affine hull. For example, the figure shows that the branches $b((RL)^k L)$, for $k \geq 0$, share the same affine hull. So in some sense the branches do 'line up' on the tree.

Now consider Figure 3.19b. This figure displays some other interesting geometrical properties. The affine hull of the branch $b(RRR)$ passes through the top of the trunk (extension of line segment AC) and bisects the angle between the trunk and $b(R)$. Thus the point of the subtree S_{RRR} that is on $b(R)$ (at D) has the same distance to the top of the trunk as does the point of the subtree S_{RRR} that is on the trunk (at B). In addition, the two triangles $\triangle ABC$ and $\triangle ADC$ are both golden gnomons. The rhombus $ABCD$ is not considered to be a 'golden rhombus', however, because that name is reserved for a rhombus whose diagonals are in the golden ratio.

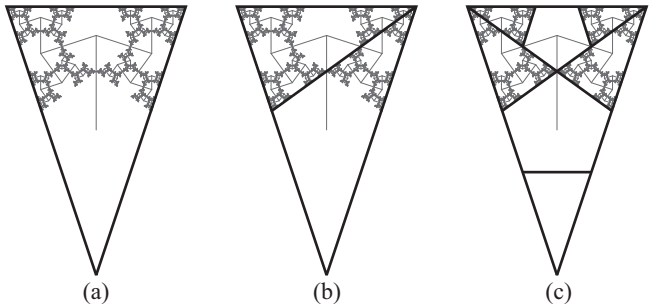


Figure 3.20. Golden triangles, golden gnomons and pentagons in $T(1/\phi, 108\,°)$

We can associate other golden triangles with $T(1/\phi, 108°)$. Figure 3.20a displays a golden triangle formed as follows. One edge goes through the top of the tree. The other two edges go through the tops of the two level one subtrees, and extend to meet at a common vertex. In Figure 3.20b, we have the same golden triangle around the tree, along with the line segment that passes through the tops of the subtrees $S_{(RL)^k LL}$ and $S_{(RL)^k RRR}$, for $k \geq 0$. This line segment divides the original golden triangle into a golden gnomon and a smaller golden triangle.

The final image in Figure 3.20c shows the original golden triangle with more line segments, showing more golden triangles and two pentagons. In this figure, the length of the base is l and the lengths of the sides are ϕl (because it is a golden triangle). When a golden triangle is divided into a golden gnomon and smaller golden triangle, the original golden triangle and smaller triangle are geometrically similar with a factor of ϕ. This can be seen using the scaling nature of the trees, because the smaller golden triangle has its base as the top of a level one subtree and the sides have length equal to the top of the actual tree (l). The sides on the larger pentagon have length l/ϕ^2 and the smaller

pentagon sides have length l/ϕ^3. The golden triangle at the bottom of the figure, with base being one side of the larger pentagon, has side lengths equal to l/ϕ. Considering the side length of the original golden triangle, we have a visual representation that

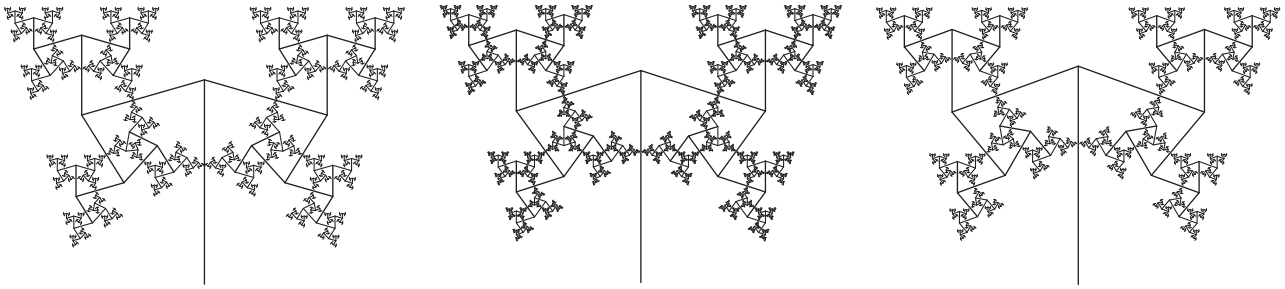
$$\frac{l}{\phi} + \frac{l}{\phi^2} + \frac{l}{\phi} = \phi l,$$

which is equivalent to

$$2\phi + 1 = \phi^3.$$

as derived in Equation 3.5.

For the sake of comparison, Figure 3.21 displays the tree $T(1/\phi, 108°)$ along with the self-contacting trees with branching angles 106° and 110°.



(a) $T(r_{sc}, 106°)$ (b) $T(1/\phi, 108°)$ (c) $T(r_{sc}, 110°)$

Figure 3.21. Comparison of $T(1/\phi, 108°)$ and two other self-contacting trees with branching angles close to 108 °

3.2.3 Golden 120

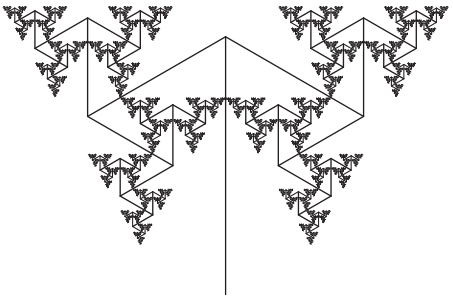


Figure 3.22. $T(1/\phi, 120°)$

The third golden self-contacting tree has branching angle 120°. This angle is not as commonly associated with the golden ratio as 108° is. However, we shall see that the appearance of equilateral triangles implies that the golden ratio also appears. As with the tree with branching angle 108°, the path to a point on S_R with minimal distance to the y-axis is $p(RR(RL)^\infty)$. Let $P_c = (x_c, y_c)$ denote the point $P_{RR(RL)^\infty}$.

Because branches of the form $b(RRR(RL)^k)$, for $k \geq 0$, are vertical if $\theta = 120°$, the calculations for this tree are simplified. The value of x_c is given by

$$\begin{aligned} x_c &= r\sin\theta + r^2\sin(2\theta)(1 + r^2 + r^4\cdots) \\ &= \frac{r\sin\theta}{1-r^2}[(1-r^2) + 2r\cos\theta] \end{aligned}$$

Substituting $\theta = 120°$:

$$x_c = \frac{r\sqrt{3}}{2(1-r^2)}[1 - r - r^2]$$

Thus r_{sc} is the root of $1 - r - r^2$ in $(0, 1)$, which of course is $1/\phi$.

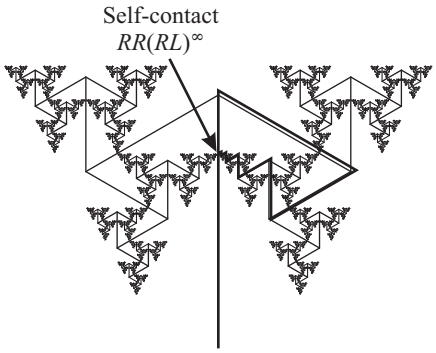
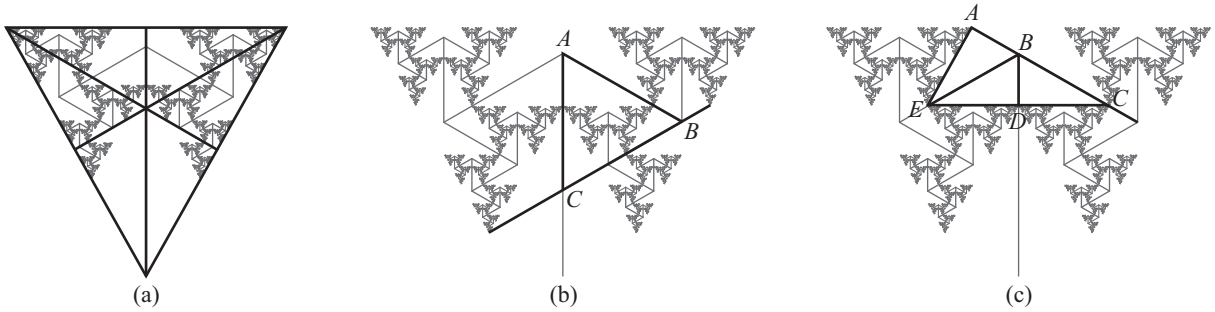


Figure 3.23. Path $p(RR(RL)^\infty)$ to self-contact on $T(1/\phi, 120°)$

One equilateral triangle that can be associated with this tree is the smallest triangle that bounds the tree, as shown in Figure 3.24a. Moreover, the medians of this triangle possess interesting features. The vertical median is coincident with the trunk (naturally). The other two medians are such that for any subtree that they intersect, the intersection is either along the trunk of the subtree, or just at one corner point of the subtree (so never crossing a branch). One of the medians shows that the branches $b((LR)^k RRL)$, for $k \geq 0$, and $b((LR)^k LLLR)$, for $k \geq 0$, have the same affine hull.



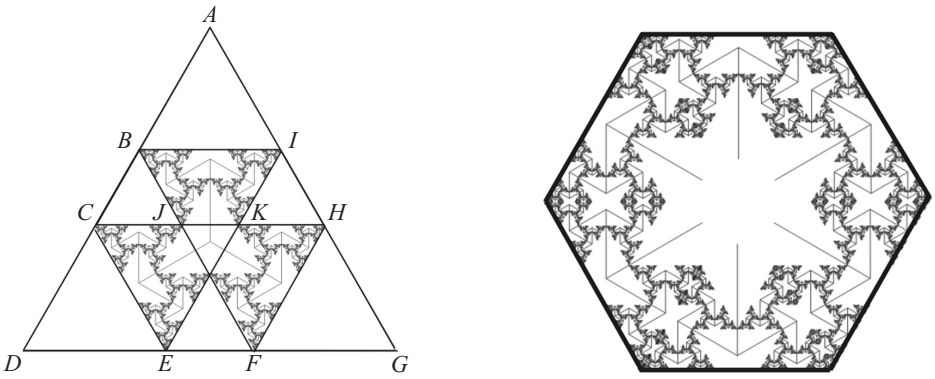
(a) (b) (c)

Figure 3.24. $T(1/\phi, 120°)$: lining up of branches and special triangles

Another equilateral triangle is displayed in Figure 3.24b. This triangle consists of the branch $b(R)$ (segment AB), the extension of the branch $b(RR)$ to the trunk (segment BC), and the portion of the trunk that joins these two line segments (segment CA). The length of each side is $1/\phi$, so the line segment that extends from $b(RR)$ divides the trunk according to the golden ratio. This same figure also shows that the point C divides the trunk according to the golden ratio (because AC has the same length as AB). The extension of CB meets the tree at two other points besides the branch of the subtree S_{RR} and its self-contact point. These other two points are both corner points of the tree.

Now consider Figure 3.24c. The line through the tip points of maximal height of the subtree S_{RRR} is horizontal, and it forms a special 30-60-90 triangle with the trunk and the branch $b(R)$. This triangle is $\triangle BDC$ in Figure 3.24c. The linear extension of the branch $b(L)$ meets the point with address $RL(LR)^\infty$ and is perpendicular to the top of the subtree S_{RLL}. Because of the left-right symmetry of the tree, $\triangle BDE$ is congruent to $\triangle BDC$. A third congruent triangle is $\triangle BAE$.

As with $T(1/\phi, 60°)$ and $T(1/\phi, 108°)$, the top tip points of $T(1/\phi, 120°)$ form a middle Cantor set with scaling dimension $\log 2/(2\log\phi)$. Another common feature of this tree with other golden trees is that the tree $T(1/\phi, 120°)$, along with two copies of itself (rotated 120° and 240° around the origin) form a figure that is self-contacting. Figure 3.25a displays equilateral triangles related to the triple tree. $\triangle ABI$ is equilateral, and each side of the triangle is l (where l denotes the length of the top of the tree). Likewise for $\triangle CDE$ and $\triangle HFG$. $\triangle BCJ$ is equilateral, with sides of length l/ϕ. Thus the line segment AD has length $l + l/\phi + l$. One can show that the ratio of AD to AC is ϕ. Thus triangles $\triangle ACH$, $\triangle BDF$ and $\triangle IEG$ are similar to $\triangle ADG$ with factor $1/\phi$. The iterated function system which consists of the three similarities that send the largest equilateral triangle $\triangle ADG$ to three triangles scaled by $1/\phi$, namely $\triangle ACH$, $\triangle BDF$ and $\triangle IEG$, corresponds to a variation on the usual Sierpiński gasket that is also self-similar [1].

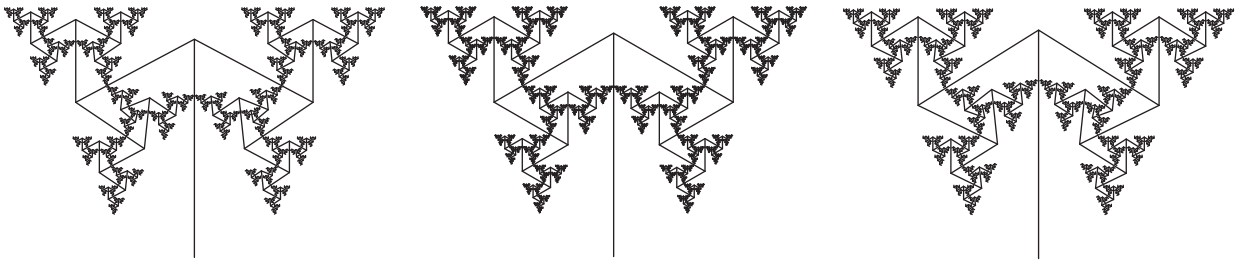


(a) (b)

Figure 3.25. $T(1/\phi, 120°)$: more equilateral triangles and a hexagon

Figure 3.25b displays a hexagon tiled with six copies of $T(1/\phi, 120°)$.

Finally, for the sake of comparison, Figure 3.26 displays the tree $T(1/\phi, 120°)$ along with the self-contacting trees with branching angles 118° and 122°.



(a) $T(r_{sc}, 118°)$ (b) $T(1/\phi, 120°)$ (c) $T(r_{sc}, 122°)$

Figure 3.26. Comparison of $T(1/\phi, 120°)$ and two other self-contacting trees with branching angles close to 120°

3.2.4 Golden 144

Finally we have the self-contacting tree with branching angle 144°. The complement of the angle 144° is 36°, which is an interior angle of both golden triangles and golden gnomons.

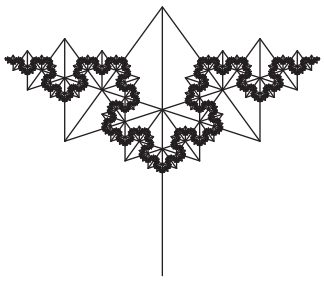


Figure 3.27. $T(1/\phi, 144°)$

First we show that the self-contacting ratio for this angle is indeed $1/\phi$. A path to a point on S_R with minimal distance to the trunk is given by $p(RR)$.

The x-coordinate of the point P_{RR} is $r \sin\theta + r^2 \sin(2\theta)$. For this point to be on the trunk, $\sin\theta + r^2 \sin(2\theta) = 0$, hence $1 + 2r\cos\theta = 0$. Solving for r_{sc} gives

$$r_{sc}(144°) = \frac{-1}{2\cos 144°} = 1/\phi.$$

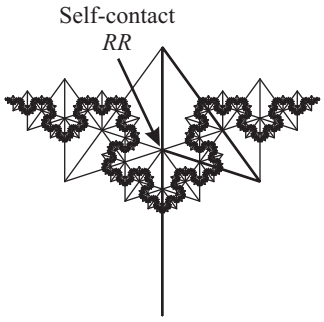


Figure 3.28. Path $p(RR)$ to self-contact on $T(1/\phi, 144°)$

The self-contact point P_{RR} divides the trunk according to the golden ratio. The point $P_{RR(LR)^\infty}$ also divides the trunk according to the golden ratio.

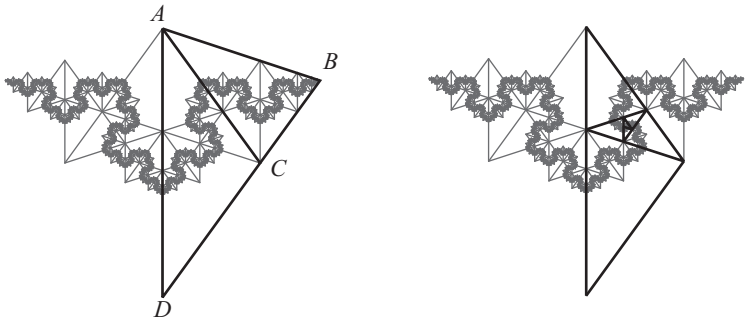


Figure 3.29. Golden figures on $T(1/\phi, 144°)$

Consider $T(1/\phi, 144°)$ in Figure 3.29a. The line segment BD through the origin (at D) and the point with address R (at C) goes through all points of the form $P_{R(LR)^k}$ for $k \geq 0$ and $P_{R(LR)^\infty}$. This line segment BD has length 1. The line segment AB goes through all the points of the form $P_{(RL)^k}$ for $k \geq 0$ and $P_{(RL)^\infty}$. The triangle $\triangle ABD$ is a golden triangle. The line segment AC that goes from the top of the trunk to the point P_R divides $\triangle ABD$ into a golden gnomon ($\triangle ACD$) and a smaller golden triangle ($\triangle ABC$).

Figure 3.29b shows branches of the form R^k. They form a spiral of golden triangles.

The tip points of $T(1/\phi, 144°)$ form a golden Koch curve. The initial line segment is shown in Figure 3.30a, with the first iteration shown in Figure 3.30b. A line segment of length l is replaced with four line segments each of length l/ϕ^2 (because they correspond to level 2 subtrees). The triangle formed from the two inner line segments (and a line segment missing from the figure) is a golden triangle because the angles are $36°, 72°, 72°$.

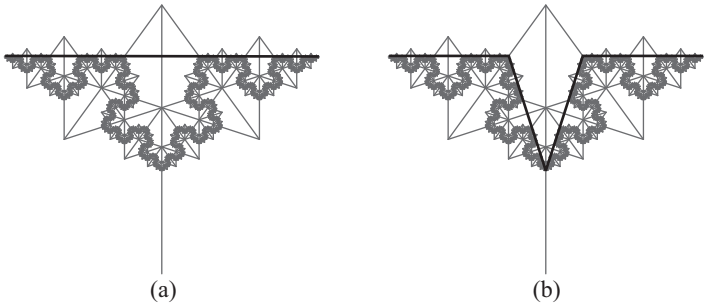


Figure 3.30. Creation of the Golden Koch Curve on $T(1/\phi, 144°)$

Figure 3.31 shows the tree along with four copies, rotated around the bottom of the trunk with angles that are multiples of 72°. The tip points of the trees form a golden Koch snowflake.

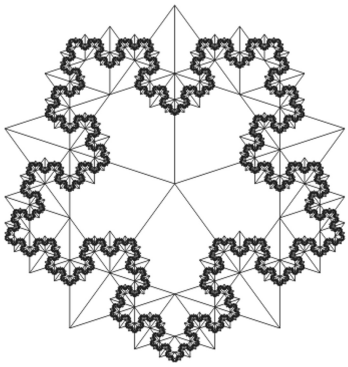
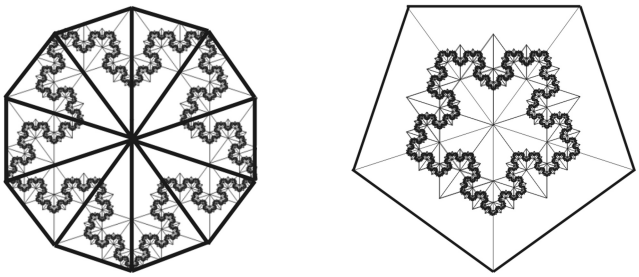


Figure 3.31. Golden Koch Snowflake with copies of $T(1/\phi, 144\,°)$

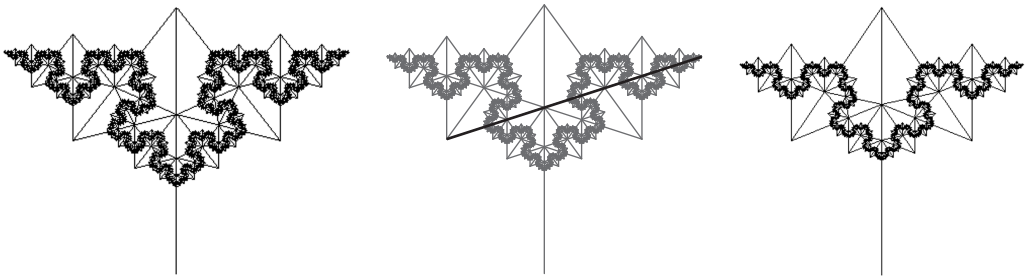
Figure 3.32a shows a decagon associated with the tree and four copies. Finally, Figure 3.32b shows a pentagon with each vertex acting as the bottom of a trunk of a tree and such that the five trees share the same point as the top of the trunk.



(a) (b)

Figure 3.32. More golden figures related to $T(1/\phi, 144\,°)$: (a) decagon and (b) pentagon

As with the other golden trees, the branches line up in a certain fashion. Figure 3.33b shows how the branches $b(RL)^k LL$ and $b(RL)^k RRR$, for $k \geq 0$, share the same affine hull. For the sake of comparison, the trees $T(r_{sc}, 142°)$ and $T(r_{sc}, 146°)$ are displayed in Figures 3.33a and c.



(a) $T(r_{sc}, 142°)$ (b) $T(1/\phi, 144°)$ (c) $T(r_{sc}, 146°)$

Figure 3.33. $T(1/\phi, 144°)$ and two other self-contacting trees with branching angles close to 144 °

3.3 Conclusions

This paper has presented four self-contacting symmetric binary fractal trees that scale with the golden ratio. The four possible branching angles are 60°, 108°, 120°, and 144°. Various geometrical figures, other fractals, and equations associated with the golden ratio can be associated with these special trees. The author hopes that the readers have enjoyed new ways to see the symmetry and beauty of the golden ratio and fractals.

Acknowledgments The author gratefully acknowledges the Natural Sciences and Engineering Research Council of Canada, the Killam Foundation, and St. Francis Xavier University for financial support. The initial research was done during doctoral research under the supervision of Dorette Pronk of Dalhousie University and interest in the golden ratio was sparked by the Halifax sculptor John MacNab. The author is also grateful for helpful comments from the reviewers.

Bibliography

[1] Broomhead, D., J. Montaldi, and N. Sidorov, "Golden gaskets: Variations on the Sierpiński sieve," *Nonlinearity*, 17 (2004), 1455–1480.

[2] Dunlap, R. A., *The Golden Ratio and Fibonacci Numbers*, World Scientific, 1970.

[3] Ghyka, M., *The Geometry of Art and Life*, Dover Publications Inc., 1977.

[4] Herz-Fischler, R., *A Mathematical History of the Golden Number*, 1st ed., Dover Publications Inc., 1998.

[5] Huntley, H. E., *The Divine Proportion*, Dover Publications Inc., 1970.

[6] Kraft, R. L., "A Golden Cantor Set," *American Mathematical Monthly*, 105 (1998), 718–725.

[7] Livio, M., *The Golden Ratio: The Story of PHI, the World's Most Astonishing Number*, Broadway Books, 2003.

[8] Mandelbrot, B., *The Fractal Geometry of Nature*, W. H. Freeman, 1982.

[9] Mandelbrot B., and M. Frame, "The Canopy and Shortest Path in a Self-contacting Fractal Tree," *The Mathematical Intelligencer*, 21(2), (1999), 18–27.

[10] Pagon, D., "Self-similar Planar Fractals Based on Branching Trees and Bushes," *Progress of Theoretical Physics Supplement*, 150 (2003), 176–187.

[11] Taylor, T. D., *Computational Topology and Fractal Trees*, Doctor of Philosophy Thesis, Dalhousie University, Canada, 2005.

[12] ——, "Homeomorphism Classes of Self-contacting Symmetric Binary Fractal Trees," *Fractals*, 15 (2007), 9–25.

[13] Walser, H., *The Golden Section*, The Mathematical Association of America, 2001.

4

Exploring Fractal Dimension, Area, and Volume

Mary Ann Connors
Westfield State University

4.1 Introduction

Fractals were conceived and introduced in the context of advanced mathematical research. However, the basic simplicity of their construction and numerous elementary representations lends itself to an introduction at the secondary school or beginning college curriculum. These fascinating geometric figures are, therefore, capable of enriching the contemporary mathematics curriculum in an important and significant manner, and, simultaneously, providing a bridge to the world of mathematics for the general public. In addition, there are numerous practical applications modeled by fractals.

In this article a fractal is considered to be a geometric shape that has the following properties:

1. The shape is self-similar,
2. The shape has fractal dimension, and
3. The shape is formed by iteration through infinitely many stages.

The word "fractal" is derived from the Latin word *fractus* meaning "broken" or "fractured". The boundary or surface of a fractal is bent, twisted, broken, or fractured.

There are many surprises in generating fractals and investigating their properties. This paper will address the notion of self-similarity, as well as fractal dimension and area and volume for fractals.

4.2 Self-Similarity

Some fractals are strictly self-similar while others are approximately or quasi self-similar. At any level of magnification of a strictly self-similar fractal there is a smaller piece of the object that is a reduced copy of the whole fractal. For example, the Sierpiński Triangle or Sierpiński Gasket (Figure 4.1) is a strictly self-similar fractal. Notice that each of the three encircled portions of Figure 4.1 is identical to the whole figure when magnified by a factor of 2. The self- similarity is recursive.

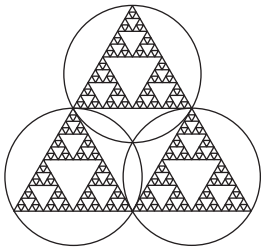


Figure 4.1.

By contrast, some fractals are nearly, or "approximately strictly" self-similar. For example, the Mandelbrot Set (Figure 4.2) is approximately or quasi self-similar. By zooming in on various parts of the Mandelbrot set, one can see smaller slightly different copies of itself, as illustrated in Figure 4.2. It is approximately self-similar mainly because of the

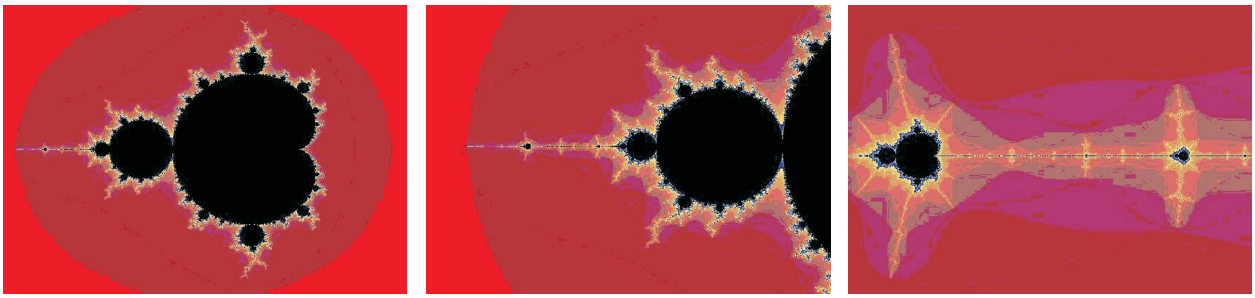


Figure 4.2.

antennae connecting the smaller copies to the main body of the set, sometimes referred to as a bug or snowman. The Mandelbrot set is generated by the iteration of complex numbers $z \to z^2 + c$ with zero as the starting point. On his website entitled The Mandelbrot Set (math.bu.edu/DYSYS/FRACGEOM/node2.html), Robert Devaney defines the Mandelbrot set as follows:

> The Mandelbrot set M consists of all complex c-values for which the corresponding orbit of 0 under $z^2 + c$ does not escape to infinity.

The images in Figure 4.2 were made with Mandelbrot Explorer on the National Technical University of Athens web page www.softlab.ntua.gr/miscellaneous/mandel/.

A video entitled "Mandelbrot Zoom" (www.youtube.com/watch?v=gEw8xpb1aRA) created by M. Eric Carr with music and lyrics by Jonathan Coulton zooms in on the Mandelbrot set for 4 minutes and 25 seconds (*YouTube*, August 12, 2006).

The Mandelbrot Set is an example of a fractal that was conceived and introduced in the context of advanced mathematical research. In this article the focus will be on strictly self-similar fractals with basic simplicity of construction and numerous elementary representations.

4.3 Sierpiński Triangle

The first fractal we will discuss is the famous Sierpiński Triangle depicted in Figure 4.3. The fractal is the result of an infinite number of iterations. There are several ways to generate it.

Method I Begin with a triangle (Level 0). Draw line segments joining the midpoints of the three sides. Remove the center triangle. The resulting figure is illustrated as Level 1 in Figure 4.3. Repeat this process (iterate) on each reduced triangle in Level 1. Iterate at each level an infinite number of times. The Sierpiński Triangle fractal is the result of an infinite number of iterations as illustrated in Figure 4.3.

An alternative way to describe the process is: Begin with a triangle (Level 0). Replace the triangle as illustrated in Figure 4.3 (Level 1) with three similar triangles such that the length of each side is $\frac{1}{2}$ the length of a side of the original. Replace each triangle in Level 1 with the three reduced copies of itself. Iterate an infinite number of times.

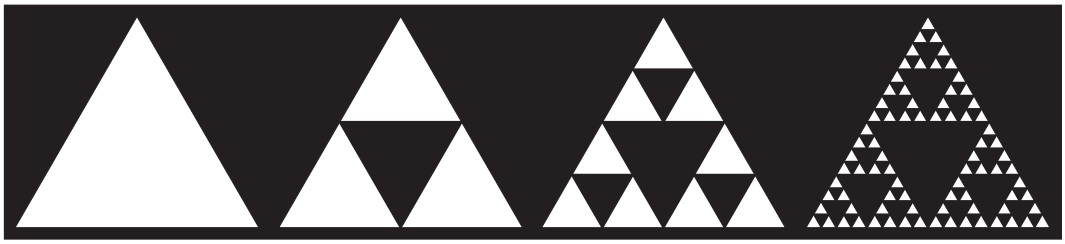


Figure 4.3.

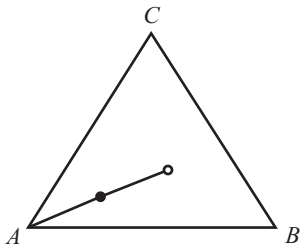


Figure 4.4.

Method II Play the Chaos Game with a triangle, a die, and the following rules:

1. Choose a random starting point (o) within the triangle ABC (Figure 4.4).
2. Toss a die with sides marked 1–6.
3. Make a dot • half the distance between the point (o) and A, when the die shows 1 or 2, between (o) and B, when the die shows 3 or 4 , or between (o) and C, when the die shows 5 or 6.
4. Use the resulting dot • and repeat steps 2–4.

After many repetitions, a geometric figure like that in Figure 4.5 is produced. It looks like a Sierpiński triangle. In fact, when the process is iterated an infinite number of times, the resulting figure is a Sierpiński triangle.

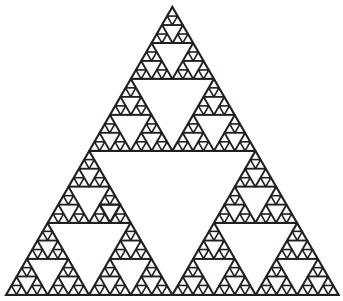


Figure 4.5.

Graphing calculators and computers generate essentially the same figure if enough steps are taken.

Method III Color the odd and even numbered regions of Pascal's Triangle illustrated in Figure 4.6 (Seymour, 1986) with two distinct colors (e.g., white for odd and black for even). Continuing ad infinitum, we once again see the Sierpiński Triangle, as shown in Figure 4.7.

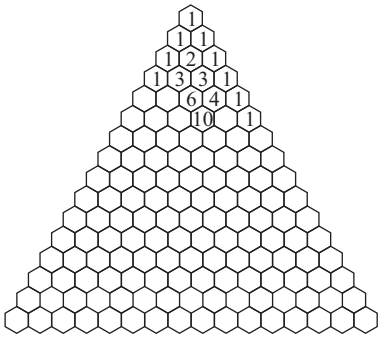


Figure 4.6.

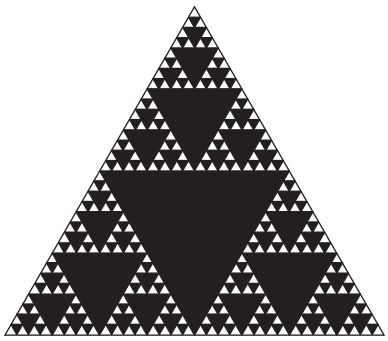


Figure 4.7.

4.4 Fractal Dimension

For an intuitive approach to a discussion on fractal dimension, consider the following example.

Disregarding the thickness of a smooth piece of aluminum foil as shown in Figure 4.8, one can regard its dimension

Figure 4.8.

to be 2. When it is crumpled up into a "solid" ball as shown in Figure 4.9, it appears to be approximating dimension 3.

Figure 4.9.

When you carefully reopen the ball of foil as pictured in Figure 4.10, it becomes an object resembling a mountain and representing a dimension between 2 and 3. A similar example compares a drawing with a sculpture and a bas relief sculpture.

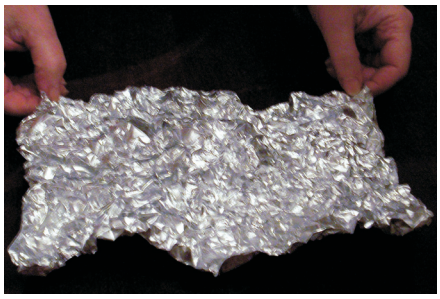


Figure 4.10.

Fractal dimension measures the degree of detail in an object or the amount of space occupied by an object. To determine the fractal dimension D of a strictly self-similar set, let N be the number of smaller pieces of the fractal that are reduced copies of the whole figure, and S the scaling factor. Then D is defined by the equation

$$N = S^D$$

For example, the Sierpiński Triangle (Figure 4.3) is strictly self-similar. The original triangle is replaced by 3 similar triangles, so $N = 3$. Each side is $\frac{1}{2}$ the size of a side of the original triangle, that is, each piece looks exactly like the original figure when magnified by a factor of 2 (scaling factor), so $S = 2$. It follows that $3 = 2^D$ and $D = \log 3/\log 2$ or $\ln 3/\ln 2$. Thus $\boldsymbol{D = 1.584}\ldots$. So the fractal dimension of the Sierpiński Triangle D is approximately 1.584.

4.5 Higher Dimension Analog of the Sierpiński Triangle: Fractal Dimension and Volume

The higher dimension analog of the Sierpiński Triangle is the Sierpiński Tetrahedron, alternately called the Sierpiński Arrowhead, or Fractal Skewed Web. See Figure 4.11 for a pictorial representation. The fractal skewed web is self-similar. It is formed by iteration through infinitely many stages and its surface is fragmented. It can be generated by beginning at Level 0 with a regular tetrahedron (one of the Platonic solids). Replace it with 4 tetrahedrons (3 on the bottom and one on top) with the length of each side $\frac{1}{2}$ the length of a side of the original to obtain Level 1. Repeat the process (iterate) by replacing each tetrahedron in Level 1 with 4 tetrahedrons with sides $\frac{1}{2}$ the length of those in Level 1. Iterate an infinite number of times.

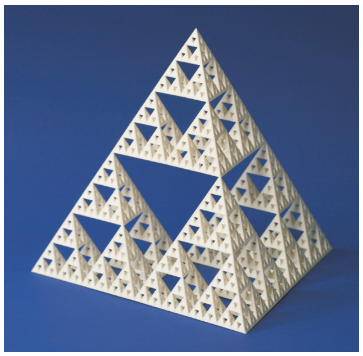


Figure 4.11. Image courtesy of George Hart, georgehart.com

Compute the fractal dimension by using $N = S^D$, to obtain $4 = 2^D$. Therefore, $\boldsymbol{D = 2}$.

Somewhat surprisingly, the dimension of the fractal skewed web is 2 after beginning with a three dimensional tetrahedron.

These examples show that fractals may have integral or non-integral dimension. Benoit Mandelbrot discusses the idea of dimension in *The Fractal Geometry of Nature* (1977, 1982, and 1983). In commenting on fractal dimension, D, Mandelbrot asserts:

> The striking fact that D need not be an integer deserves a terminological aside. If one uses *fraction* broadly, as synonymous with a noninteger (sic) real number, several of the above listed values of D are fractional, and indeed the Hausdorff Besicovitch dimension is often called *fractional dimension*. But D may be an integer. . . I call D a fractal dimension. [12, p. 15]

If you are curious about what must be removed from the solid tetrahedron to produce Level 1, you can explore with play dough or tape equilateral triangles inside Level 1 until you form the shell of the solid that would lie inside. I suggest not telling the answer. Let students investigate. They will be very surprised to see that it is another Platonic solid, an octahedron.

Another question to pose is "What is the volume of the Fractal Skewed Web?" First, consider the volume of the octahedron (Figure 4.12) that has been removed from the original tetrahedron.

Begin with a regular octahedron (Figure 4.12) with edges of length one unit. Then $AB = 1$ unit (as do AD, DC, and PD), $DE = 1/2$, $QE = 1/2$, $PE = \sqrt{3}/2$, and $PQ = \sqrt{2}/2$. Since the area of the base of the square pyramid $ABCD = 1$ square unit and the height is $\sqrt{2}/2$ units, the volume of the square pyramid

$$ABCDP = \left(\frac{1}{3}\right)(1)\left(\frac{\sqrt{2}}{2}\right) = \frac{\sqrt{2}}{6}$$

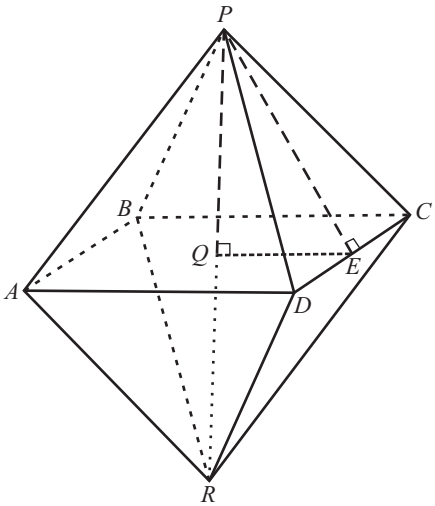


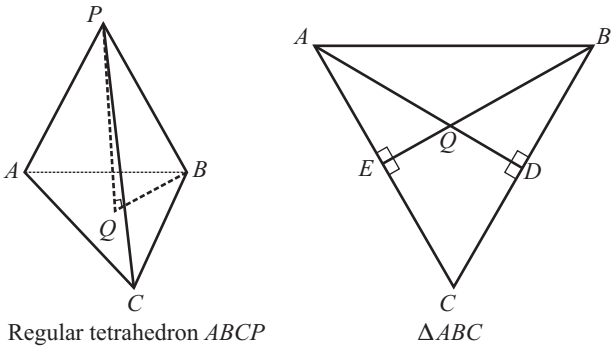
Figure 4.12.

cubic units, and the volume of the regular octahedron $= 2\sqrt{2}/6$ or $\sqrt{2}/3$ cubic units.

Given a regular tetrahedron $ABCP$ (Figure 4.13) with edges of length one unit, it follows that $AB = BC = AC = 1$, $BD = 1/2$, and $AD = \sqrt{3}/2$. The three medians of a triangle meet at a point two-thirds of the distance from the vertex to the midpoint of the opposite side. Since the altitudes of the equilateral triangle ABC (Figure 4.13) coincide with the medians,

$$BQ = \left(\frac{2}{3}\right)\left(\frac{\sqrt{3}}{2}\right) = \frac{\sqrt{3}}{3} \quad \text{and}$$

$$DQ = \left(\frac{1}{3}\right)\left(\frac{\sqrt{3}}{2}\right) = \frac{\sqrt{3}}{6}.$$



Regular tetrahedron $ABCP$ ΔABC

Figure 4.13.

(Recall $BP = 1$ unit.) Since $BP^2 = BQ^2 + PQ^2$, we have $PQ = \sqrt{6}/3$.

Since the area of the base of the regular tetrahedron $ABC = \sqrt{3}/4$ square units and the height of the tetrahedron $PQ = \sqrt{6}/3$ units,

$$\text{the volume of the regular tetrahedron } (ABCP) = \left(\frac{1}{3}\right)\left(\frac{\sqrt{3}}{4}\right)\left(\frac{\sqrt{6}}{3}\right) = \frac{\sqrt{2}}{12} \text{ cubic units.}$$

The volume of the octahedron is equal to four times that of the tetrahedron when all edges are one unit long.

When the Fractal Skewed Web is generated, we see that at each level one half of the previous volume is removed. For example, at Level 1 there are four tetrahedrons left when an octahedron with the same edge length is removed. The sum of the infinite geometric series represents the volume removed from the original figure.

Let the volume of the Fractal Skewed Web at Level 0 (regular tetrahedron) $= V$. The *volume* which has been removed from the regular tetrahedron after the Fractal Skewed Web is completely generated,

$$\sum_{n=1}^{\infty} \frac{1}{2^n} V = \frac{1}{2}V + \frac{1}{4}V + \frac{1}{8}V + \frac{1}{16}V + \cdots = V$$

(the entire volume of the original tetrahedron). Therefore, the volume of the Fractal Skewed Web equals zero.

In summary, the dimension of the fractal skewed web is equal to 2 after beginning with the three dimensional tetrahedron with volume $\sqrt{2}/12$ cubic units. And although a 2-dimensional figure is expected to have 0 volume, this one is counter-intuitively 2-dimensional.

4.6 Other Surprising Fractals of Dimension 2

Example 1. The **Harter-Heighway Dragon** (Figures 4.14 and 4.15) is sometimes called the Paper Folding Dragon. A picture of a different iteration of the dragon appears at the beginning of each chapter of Michael Crichton's 1990 publication of *Jurassic Park*.

Begin with a line segment (Level 0). Replace it with two segments scaled to a ratio of $1/\sqrt{2}$ to form a right angle positioned so that each of the segments of the previous iteration would be the hypotenuse of a right triangle. The right

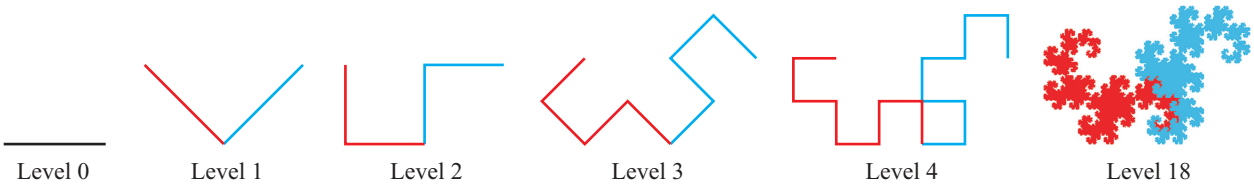


Figure 4.14. These images were generated using the "Paper Folding" software, courtesy of Heinz Otto Peitgen, distributed at The Chaos, Fractals, and Dynamics Symposium III, Princeton Plasma Physics Laboratory, July 31–August 5, 1995.

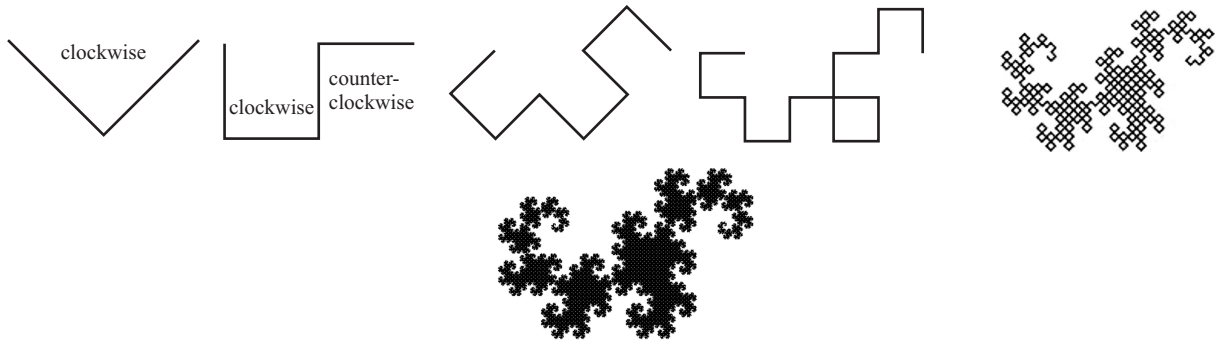


Figure 4.15.

angles are rotated 45° alternatively to the right (clockwise) and to the left (counterclockwise) as illustrated in Figures 4.14 and 4.15.

Color is needed to see the **strict** self-similarity. Otherwise the dragon looks like Figure 4.15.

Compute the fractal dimension of the Harter-Heighway Dragon by using $N = S^D$ to obtain $2 = (2^{1/2})^D$, and thus $\boldsymbol{D = 2}$. The Harter-Heighway Dragon is space-filling curve with a fractal boundary.

Example 2. Peano's Space-Filling Curve (Figure 4.16)

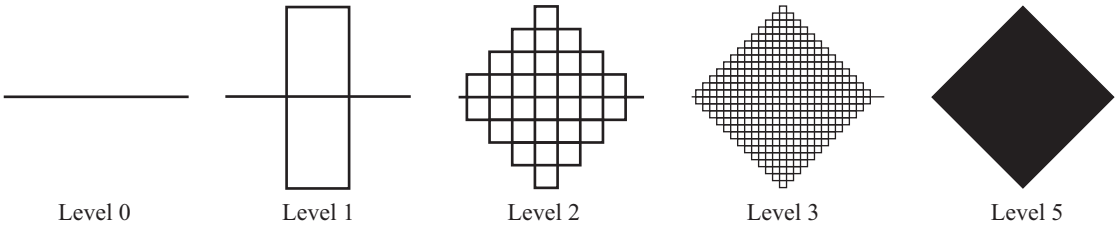


Figure 4.16.

Begin with a line segment (Level 0). Replace the line segment with nine segments $1/3$ the size of the previous segment as illustrated in Level 1. Repeat the process with respect to each new line segment, and continue the process an infinite number of times.

Compute the fractal dimension using $N = S^D$ to obtain $9 = 3^D$ and thus $\boldsymbol{D = 2}$.

One can consider Peano's Space-Filling Curve, whose image is a "filled out" square, as a fractal if one considers it as a curve [17, p. 95].

4.7 Sierpiński Carpet

A small change in the generator of the Peano Curve produces a different fractal with a different dimension. If at each stage we replace a line segment with 8 miniature line segments rather than 9 miniature line segments, then the resulting fractal (Sierpiński Carpet, Figures 4.17 and 4.18) has dimension $1.892\ldots$, rather than the fractal (Peano curve) with dimension 2.

The Sierpiński Carpet can also be generated in more than one way.

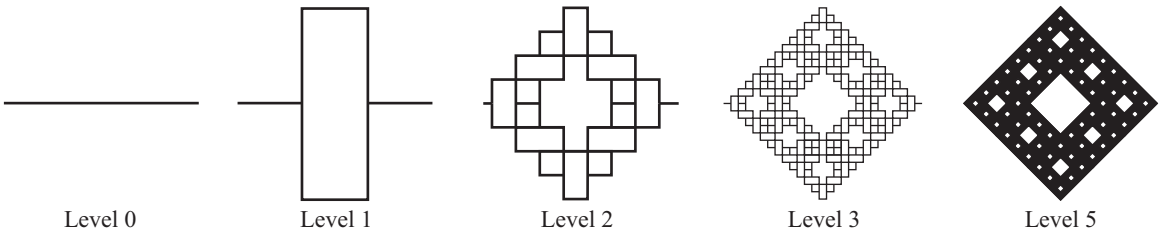


Figure 4.17.

Figure 4.18.

Method I Begin with a line segment (Level 0). Replace the line segment with eight segments $1/3$ the size of the previous segment as illustrated in Level 1 (Figure 4.17). Iterate an infinite number of times.

Method II Another method of generating the Sierpiński Carpet is to begin with a solid square at Level 0. For the first iteration replace the square with eight squares with sides 1/3 the length of the previous square, leaving the center empty as illustrated in Figure 4.18. Iterate an infinite number of times.

To compute the fractal dimension use $N = S^D$ to obtain $8 = 3^D$. Thus $D = \log 8/\log 3$, so that $\boldsymbol{D = 1.892}\ldots$.

4.8 Higher Dimension Analog of the Sierpiński Carpet

The higher dimension analog of the Sierpiński Carpet is the *Menger Sponge* (Figure 4.19). Begin with a solid cube. Replace with cubes with sides 1/3 the length of the previous cube leaving out the center on each face and the center of the cube. Iterate an infinite number of times.

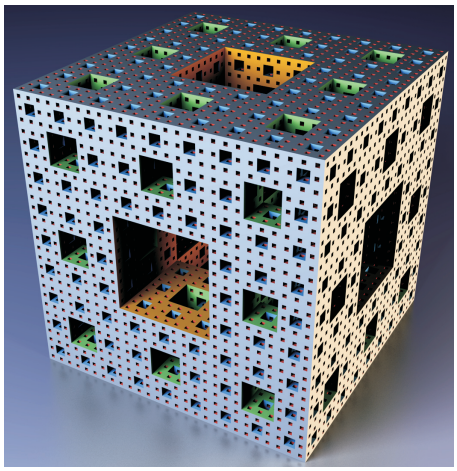


Figure 4.19. Menger Sponge created by Niabot. commons.wikimedia.org/wiki/File:Menger-Schwamm-farbig.png

Use $N = S^D$ to obtain $20 = 3^D$. Thus $D = \log 20/\log 3$, so the fractal dimension is $\boldsymbol{D = 2.722}\ldots$.

4.9 Koch Snowflake: Fractal Dimension, Perimeter, and Area

Much has been written about the Koch Snowflake (Figure 4.20). Begin with an equilateral triangle with sides of length one unit. Replace each side with four segments one third the length of the side of the original triangle forming an equilateral triangle with the base removed in the center third. Repeat an infinite number of times. With $N = 4$ and $S = 3$, we find that the dimension is $\boldsymbol{D = 1.261}\ldots$. The snowflake has infinite perimeter,

$$P = \sum_{n=1}^{\infty} 3\left(\frac{4}{3}\right)^{n-1} = \infty.$$

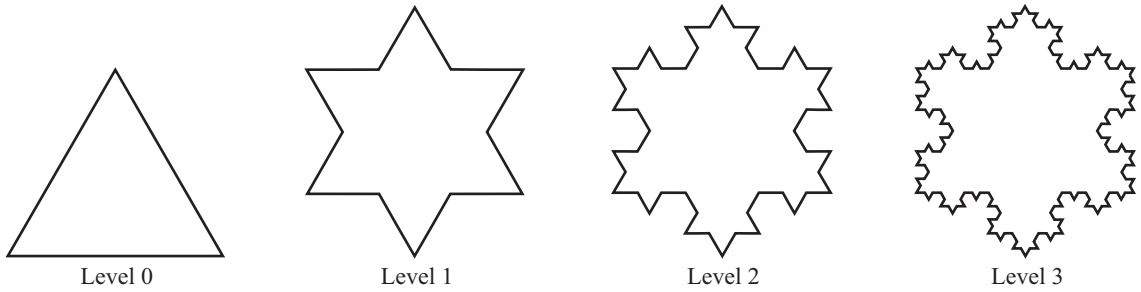


Figure 4.20.

But its finite area is equal to $\frac{8}{5}$ times the area of the original triangle at Level 0:

$$A = \frac{\sqrt{3}}{4} + \frac{\sqrt{3}}{4}\sum_{n=1}^{\infty}\frac{1}{3}\left(\frac{4}{9}\right)^{n-1} = \frac{2\sqrt{3}}{5} = \frac{8}{5}\left(\frac{\sqrt{3}}{4}\right) \quad \text{square units.}$$

James Sandefur [22, p. 113] presents another method of calculating the area of the Koch Snowflake using its self-similarity property. Consider the snowflake itself (Figure 4.21). Let A represent the area of the original triangle and x equal the area between *one* side of the original triangle and the Koch curve. Then the area of the Koch Snowflake is $A + 3x$. The area x equals the area of the middle triangle plus the area of the four congruent parts around the triangle. Since the middle triangle is similar to the original triangle, and its sides are $\frac{1}{3}$ the length of the original, the area of the middle triangle is $\frac{1}{9}A$. The four congruent parts around the middle triangle will each have area $\frac{1}{9}x$, since the region is similar to the region with area x and reduced by a factor of $\frac{1}{3}$. It follows that $x = \frac{1}{9}A + \frac{4}{9}x$. Solving for x gives $x = \frac{1}{5}A$. Hence, the area of the Koch Snowflake is

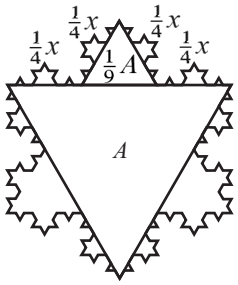


Figure 4.21.

$$A + 3x = A + \frac{3}{5}A = \frac{8}{5}A = \frac{8}{5}\left(\frac{\sqrt{3}}{4}\right) = \frac{2\sqrt{3}}{5} \quad \text{square units.}$$

4.10 Higher Dimension Analog of the Koch Snowflake: Surface Area, and Volume

One version of a higher dimension analog of the Koch Snowflake [2] is illustrated in Figure 4.22.

To generate this fractal begin with a tetrahedron with each side of length one unit. Divide each face into four equilateral triangles. On the center triangle add a tetrahedron with each side of length $\frac{1}{2}$ unit. Repeat an infinite number of times adding tetrahedrons with sides $\frac{1}{2}$ the length of the previous iteration. In order to generate this fractal, each face of the regular tetrahedron is divided into four equilateral triangles. A new tetrahedron is added on the center triangle. As a result, one of the four congruent equilateral triangles is covered; however, three new ones are added.

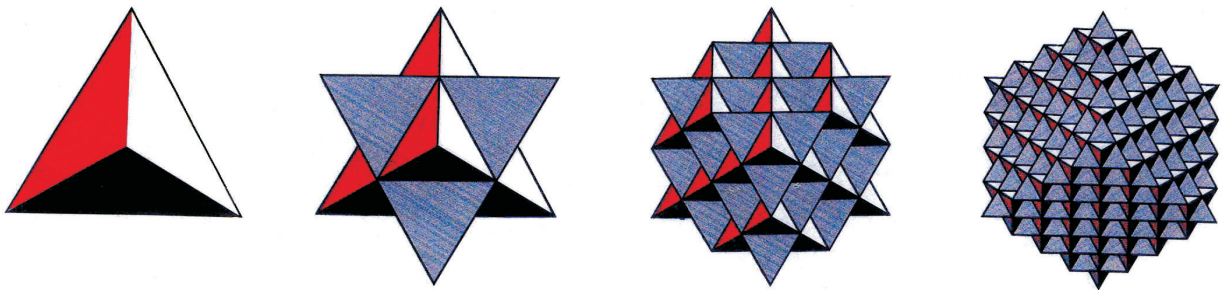


Figure 4.22.

Therefore, each face contains six triangles, resulting in a ratio of the six triangles to the previous four. It follows that the surface area

$$SA = \sum_{n=1}^{\infty} \sqrt{3}\left(\frac{3}{2}\right)^{n-1} = \infty.$$

While the surface area is infinite, the volume is finite. The volume

$$V = \frac{\sqrt{2}}{12} + \frac{\sqrt{2}}{12}\sum_{n=1}^{\infty} \frac{1}{2}\left(\frac{3}{4}\right)^{n-1} = \frac{\sqrt{2}}{4} \quad \text{cubic units.}$$

It is interesting and perhaps not surprising that its volume is the same as the volume of its limiting cube. The edge length of the limiting cube is $\sqrt{2}/2$ units and the diagonal of each face is one unit in length.

An alternate method of using self-similarity produces the same result. Let V equal the volume of the original tetrahedron and let x represent the volume between *one* face of the original tetrahedron and the three-dimensional Koch curve analog. Then the volume of the higher dimensional Koch Snowflake is $V + 4x$. The volume x equals the volume of the miniature tetrahedron on the face plus the volume of the four congruent parts around the original tetrahedron. Since the miniature tetrahedron is similar to the original tetrahedron, and its sides are $\frac{1}{2}$ the length of the original, the volume of the miniature tetrahedron is $\frac{1}{8}V$. The four congruent parts around the original tetrahedron will each have volume $\frac{1}{8}x$, since the region is similar to the region with area x and reduced by a factor of $\frac{1}{2}$. It follows that $x = \frac{1}{8}V + \frac{4}{8}x$. Solving for x gives $x = 2V$. Hence, the volume of the three-dimensional Koch Snowflake analog is

$$V + 4x = V + 2V = 3V = 3\left(\frac{\sqrt{2}}{12}\right) = \frac{\sqrt{2}}{4} \quad \text{cubic units.}$$

These examples span the breadth of a large number of mathematical topics at a variety of levels in secondary school and college including algebra, geometry, discrete mathematics, and calculus. Note in particular the interesting application of logarithms in computing fractal dimension. Generating the Mandelbrot and Julia Sets requires understanding of complex numbers and trigonometry. It is interesting to note that Mitsuhiro Shishikura proved that the Hausdorff dimension of the **boundary** of the Mandelbrot set also equals **2** [24, pp. 225–267].

It is important to mention that there are various methods of computing several types of dimension related to fractals. In *Chaos and Fractals: New Frontiers of Science* (1992), Heinz-Otto Peitgen, Hartmut Jurgens, and Dietmar Saupe focus on special forms of Mandelbrot's *fractal dimension*: self-similarity dimension, compass (divider) dimension, and box-counting dimension. Peitgen, Jurgens, and Saupe declare:

> Dimension is not easy to understand. At the turn of the century it was one of the major problems in mathematics to determine what dimension means and which properties it has... And since then it has become somewhat worse because mathematicians have come up with some ten different notions of dimension: topological dimension, Hausdorff dimension, fractal dimension, self-similarity dimension, box-counting dimension, capacity dimension, information dimension, Euclidean dimension, and more, [17, p. 202]

There are various practical applications of fractals. For example, fractal geometry is used to model complex forms such as plants, weather, human body organs and rhythms, economic and socioeconomic patterns, music, ecosystem dynamics, planetary orbits, galaxy clusters, geological activity, and dynamical diseases. Fractal dimension can measure the texture and complexity of irregular shapes such as coastlines, mountains, and clouds. The fractal dimension of the surface of a metal could be a valuable measurement of the metal's strength. 3D fractal dimension also has been used to determine the surface roughness of erythromycin acistrate tablets [20]. Fractals are used in movie making to store images in much smaller amounts of space than otherwise required. In addition to extensive practical applications, beautiful pictures, artwork, and even fabric patterns for clothing are created by generating fractal designs.

Fractal geometry provides a different perspective to view the world than Euclidean and other geometries. Fractal geometry is exciting mathematics, providing many opportunities for learners to make their own discoveries and to delight in its surprises.

Bibliography

[1] Briggs, John, *Fractals: the Patterns of Chaos*, New York: Touchstone, 1992.

[2] Camp, Dane R., "A Fractal Excursion," *Mathematics Teacher* LXXXIV (4), (April 1991), 265–75.

[3] Carr, M. Eric and Jonathan Coulton, "Mandelbrot Zoom", *You Tube*, Mandelbrot Set Zoom, August 12, 2006. www.youtube.com/watch?v=gEw8xpb1aRA

[4] Connors, Mary Ann and Anna Rose Haralampus, *Exploring Fractal Dimension*, Unpublished booklet, 1991.

[5] ——, *Exploring Fractals: From Cantor Dust to the Fractal Skewed Web*, Unpublished booklet, 1994, 1995.

[6] Connors, Mary Ann, *Exploring Fractals*, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004. 2005, 2006, 2007, 2008, 2009. www.math.umass.edu/~mconnors/fractal/fractal.html

[7] Crichton, Michael, *Jurassic Park*, New York: Ballentine Books, 1990.

[8] Devaney, Robert L., *Chaos, Fractals and Dynamics*, Menlo Park: Addison-Wesley, 1990.

[9] ——, *The Mandelbrot Set*. math.bu.edu/DYSYS/FRACGEOM/node2.html

[10] Gleick, James, *Chaos: Making a New Science*, New York: Viking, 1987.

[11] Jacobs, Harold R., *Mathematics: A Human Endeavor*, New York: Freeman, 1982.

[12] Mandelbrot, Benoit, *The Fractal Geometry of Nature*, New York: W. H. Freeman and Company, 1983.

[13] McGuire, Michael, *An Eye For Fractals*, Redwood City: Addison-Wesley, 1991.

[14] National Technical University of Athens, *Mandelbrot Explorer*, 1994. www.softlab.ntua.gr/miscellaneous/mandel/

[15] Oliver, Dick, *Fractal Vision*, Carmel, IN: SAMS Publishing, 1992.

[16] Peitgen, Heinz-Otto et al., *Fractals for the Classroom: Strategic Activities Volume One*, New York: Springer-Verlag, 1991.

[17] Peitgen, Heinz-Otto et al., *Chaos and Fractals: New Frontiers of Science*, New York: Springer-Verlag, 1992.

[18] Peitgen, Heinz-Otto and Dietmar Saupe (eds.), *The Science of Fractal Images*, New York: Springer-Verlag, 1988.

[19] Peterson, Ivars, *The Mathematical Tourist*, New York: Freeman, 1988.

[20] Riippi, Merja et al., *European Journal of Pharmaceutics and Biopharmaceutics*, 46(3), (November, 1998), 339–345.

[21] Rucker, Rudy, *Mind Tools*, Boston: Freeman, 1987.

[22] Sandefur, James. T., "Using Self-Similarity to Find Length, Area, and Dimension," *The American Mathematical Monthly*, 103(2), (Feb. 1996), 107–120.

[23] Seymour, Dale, *Visual Patterns in Pascal's Triangle*. Palo Alto, CA: Dale Seymour Publications, 1986.

[24] Shishikura, Mitsuhiro, "The Hausdorff Dimension of the Boundary of the Mandelbrot Set and Julia Sets," *Annals of Mathematics*, 2nd Series 147(2), (March, 1998), 225–267.

[25] Steen, Lynn Arthur, *On the Shoulders of Giants: New Approaches to Numeracy*, Washington, DC: National Academy Press, 1990.

[26] Stevens, Roger T., *Creating Fractals*, Hingham, MA: Charles River Media, 2005.

[27] Stewart, Ian, "Les Fractals," *Les Chroniques de Rose Polymath*, Paris: Belin, 1982.

5

Points in Sierpiński-like Fractals

Sandra Fillebrown, Joseph Pizzica, and Vincent Russo
Saint Joseph's University

Scott Fillebrown
University of Maryland

The Sierpiński Triangle (see Figure 5.1) is one of the most recognized fractals. It has many well-known properties and there are many different ways to define it. For example, one can define the Sierpiński Triangle as what is left after removing certain sets of points; see, for example, [2, p. 180]. Another definition of the Sierpiński Triangle, and the one we will use, is that the Sierpiński Triangle is the fixed point of an Iterated Function System (IFS). See [1] for a complete introduction to Iterated Function Systems; a brief description covering what is needed for our purposes is given below. Defined this way, an interesting question is exactly which points (x, y) are in the triangle since the IFS definition does not make it readily apparent. However, it is not too difficult to show that for the version of the Sierpiński Triangle that has corners at $(0, 0)$, $(0, 1)$ and $(1, 0)$, the points (x, y) in the Sierpiński Triangle are those points that have a binary expansion with a certain property. Specifically, points (x, y) that have binary expansions such that x and y do not both have a 1 in the same position are in the Sierpiński Triangle [9]. In this paper, we extend this idea to the whole class of "Sierpiński-like" fractals, developing a method for generating similar rules for which points (x, y) are or are not in the fractal.

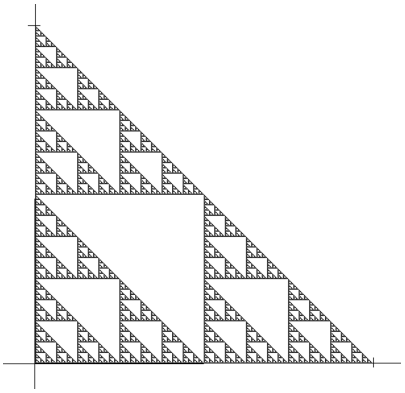


Figure 5.1. The Sierpiński Triangle

By Sierpiński-like fractal, we mean fractals that can be generated by an Iterated Function System (IFS) that consists of three functions f_0, f_1 and f_2 where each f_i is a scaling by $\frac{1}{2}$ and may include horizontal or vertical reflections (or both) and rotations of integer multiples of $\frac{\pi}{2}$ including none. We adopt the convention of describing all transformations as having no rotation or a rotation of $\frac{\pi}{2}$, followed by zero, one or two reflections. We will use the following numbering for these transformations:

$$M_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \text{ no rotation, no reflections}$$

$$M_2 = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \text{ no rotation, reflection across } y\text{-axis}$$

$$M_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \text{ no rotation, reflection across } x\text{-axis}$$

$$M_4 = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}, \text{ no rotation, reflection across both axes}$$

$$M_5 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, \text{ rotation by } \frac{\pi}{2}, \text{ no reflections}$$

$$M_6 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \text{ rotation by } \frac{\pi}{2} \text{ followed by a reflection across the } y\text{-axis}$$

$$M_7 = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}, \text{ rotation by } \frac{\pi}{2} \text{ followed by a reflection across the } x\text{-axis}$$

$$M_8 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \text{ rotation by } \frac{\pi}{2} \text{ followed by a reflection across both axes}$$

Thus, the IFS's of interest to us are those where each f_i can be written in the form

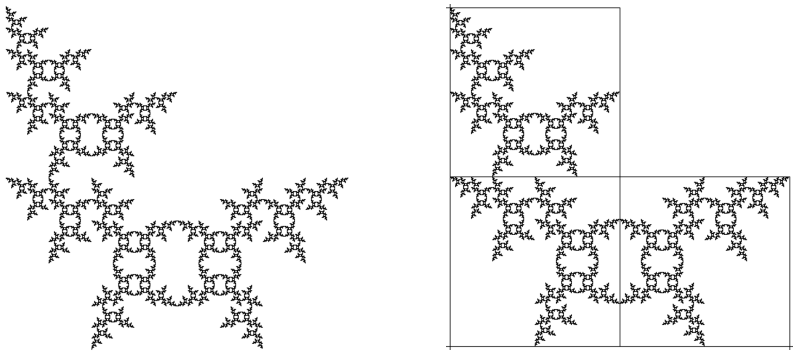
$$f_i \begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{2} M_j \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_{x_i} \\ t_{y_i} \end{pmatrix}, \quad i = 0, 1, 2.$$

If we let Q be the unit square $[0, 1] \times [0, 1]$ and label the three relevant "quadrants" so that $Q_0 = \left[0, \frac{1}{2}\right] \times \left[0, \frac{1}{2}\right]$, $Q_1 = \left[\frac{1}{2}, 1\right] \times \left[0, \frac{1}{2}\right]$ and $Q_2 = \left[0, \frac{1}{2}\right] \times \left[\frac{1}{2}, 1\right]$, then for each f_i we choose the translation $t_i = \binom{t_{x_i}}{t_{y_i}}$ so that $f_i \binom{x}{y} \in Q_i$ for all $(x, y) \in Q$. That is, f_0 maps the unit square to the lower left quadrant, f_1 to the lower right and f_2 to the upper left. The combination of i and M_j uniquely determine t_i for each i. The fractal associated with such an IFS is the unique fixed point of the IFS in the space of compact sets in the plane under the Hausdorff metric. That is, the fractal associated with the IFS consisting of f_0, f_1 and f_2 is the unique compact set A such that

$$f_0(A) \cup f_1(A) \cup f_2(A) = A$$

where by $f_i(A)$ we mean $\{f_i(x, y) \mid (x, y) \in A\}$. Figure 5.1 shows an example of such a Sierpiński-like fractal. For more background on Iterated Functions Systems see [1], [5] or [8]. In particular, pictures of all such Sierpiński-like fractals up to symmetry can be found in [8].

We will adopt the notation of naming our fractals by the subscripts of the matrices M_j that define f_0, f_1 and f_2. Thus, if we use M_7, M_8 and M_1 for f_0, f_1 and f_2 respectively, we will call this fractal 781. Figures 5.2a and 5.2b show fractal 781 both with and without lines marking the four quadrants of the unit square. For this fractal, in the lower left quadrant of the unit square we see a scaled version of the fractal that has been rotated by $\frac{\pi}{2}$, reflected vertically (transformation M_7) and then translated; in the lower right quadrant we see a scaled version that has been rotated by $\frac{\pi}{2}$, reflected across both axes (transformation M_8) and then translated; and in the upper left quadrant we see a scaled



(a) (b)

Figure 5.2. Fractal 781

version with no reflections or rotations (transformation M_1) and translated. The specific functions for fractal 781 are:

$$f_0\begin{pmatrix}x\\y\end{pmatrix} = \frac{1}{2}\begin{pmatrix}0 & -1\\-1 & 0\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix} + \begin{pmatrix}\frac{1}{2}\\\frac{1}{2}\end{pmatrix}$$

$$f_1\begin{pmatrix}x\\y\end{pmatrix} = \frac{1}{2}\begin{pmatrix}0 & 1\\-1 & 0\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix} + \begin{pmatrix}\frac{1}{2}\\\frac{1}{2}\end{pmatrix}$$

$$f_2\begin{pmatrix}x\\y\end{pmatrix} = \frac{1}{2}\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix} + \begin{pmatrix}0\\\frac{1}{2}\end{pmatrix}$$

In general, we will refer to fractal PQR as the fractal obtained by using M_P, M_Q, and M_R.

Although we know that the fractal is a well-defined compact set and there are methods of quickly generating pictures of the fractal (for example, see [7] or [10] to download free software), we are interested in a way of deciding whether a specific point (x, y) is or is not in the given fractal. In what follows, we describe such an algorithm.

We begin by writing the point of interest in binary:

$$x = .x_1x_2x_3x_4\ldots$$

$$y = .y_1y_2y_3y_4\ldots$$

where

$$x = \sum_{k=1}^{\infty}\frac{x_k}{2^k} \qquad \text{and} \qquad y = \sum_{k=1}^{\infty}\frac{y_k}{2^k}.$$

We note that this representation is not always unique; in particular, any number $a = \frac{k}{2^N}$, where k is odd and less than 2^N, can be written as $a = .a_1a_2a_3a_4\ldots$ in binary such that $a_N = 1$ and $a_j = 0$ for all $j \geq N+1$. However, a can also be written with $a_N = 0$ and $a_j = 1$ for all $j \geq N+1$. In what follows, when we say that a point (x, y) *has* a binary representation with certain properties, it may be the case that if x or y is $\frac{k}{2^N}$ where k is odd, then its representation ending in all $0's$ may have the desired property while the representation ending in all $1's$ does not, or vice versa. As long as one or the other representation (or both) has the desired property, then we will be satisfied. In what follows we will call x_k and y_k the kth bit-pair of the point (x, y) and denote it by $[x_k, y_k]$.

Our method for determining whether a point (x, y) is in a particular fractal PQR uses the sequence of bit-pairs of the point (x, y). For any given fractal, we will generate a "rule table". This table consists of eight states, labelled $S_1, S_2 \ldots S_8$ and a special *Fail* state. The *Fail* state is reached precisely when the point (x, y) is not in the fractal. The "rules" are just a function that maps states and bit-pairs to states. Thus, for each state, there are four rules, one for each of the possible bit-pairs $[0, 0]$, $[0, 1]$, $[1, 0]$ and $[1, 1]$. Further, exactly one bit-pair per state must map to the *Fail* state, i.e., represent a point that is not in the fractal; this is a consequence of the IFS mapping to three out of four of the quadrants.

If we let $r\,(S_i, [a, b])$ denote our rules, we construct the following algorithm:

- Let $T_1 = S_1$; that is, begin in state S_1.
- While $T_{k+1} \neq \textit{Fail}$, let $T_{k+1} = r\,(T_k, [x_k, y_k])$; that is, process the bit-pairs of (x, y) moving from state to state while not "failing."

Then the points (x, y) that are in the fractal are precisely those points that have a binary representation that never lands in the *Fail* state.

The rules of the table are determined by the three transformations M_P, M_Q and M_R that make up the fractal PQR. Each state S_i corresponds to the transformation M_i with the same subscript. The value of $r\,(S_i, [a, b])$ will be either *Fail* or the state associated with the transformation that is the result of taking the product of M_i with one of M_P, M_Q or M_R. The following table shows exactly which product is used for each state and bit-pair combination:

State S_1	State S_2	State S_3	State S_4
$S_1, [0,0] : M_1M_P$	$S_2, [0,0] : M_2M_Q$	$S_3, [0,0] : M_3M_R$	$S_4, [0,0] : \textit{Fail}$
$S_1, [1,0] : M_1M_Q$	$S_2, [1,0] : M_2M_P$	$S_3, [1,0] : Fail$	$S_4, [1,0] : M_4M_R$
$S_1, [0,1] : M_1M_R$	$S_2, [0,1] : \textit{Fail}$	$S_3, [0,1] : M_3M_P$	$S_4, [0,1] : M_4M_Q$
$S_1, [1,1] : \textit{Fail}$	$S_2, [1,1] : M_2M_R$	$S_3, [1,1] : M_3M_Q$	$S_4, [1,1] : M_4M_P$
State S_5	State S_6	State S_7	State S_8
$S_5, [0,0] : M_5M_R$	$S_6, [0,0] : M_6M_P$	$S_7, [0,0] : \textit{Fail}$	$S_8, [0,0] : M_8M_Q$
$S_5, [1,0] : M_5M_P$	$S_6, [1,0] : M_6M_R$	$S_7, [1,0] : M_7M_Q$	$S_8, [1,0] : \textit{Fail}$
$S_5, [0,1] : \textit{Fail}$	$S_6, [0,1] : M_6M_Q$	$S_7, [0,1] : M_7M_R$	$S_8, [0,1] : M_8M_P$
$S_5, [1,1] : M_5M_Q$	$S_6, [1,1] : \textit{Fail}$	$S_7, [1,1] : M_7M_P$	$S_8, [1,1] : M_8M_R$

Table 5.1. Rule Table for Fractal PQR

Since each matrix product gives a matrix that is one of the eight transformations and each transformation is associated with the corresponding state, the above table gives the "rules" for moving from one state to the next when processing the bit-pairs of the point (x, y).

We illustrate the above with an example. Below is the rule table for fractal 781 and the results of processing several points.

State S_1	State S_2	State S_3	State S_4
$[0,0] \to S_7$	$[0,0] \to S_7$	$[0,0] \to S_3$	$[0,0] \to \textit{Fail}$
$[1,0] \to S_8$	$[1,0] \to S_8$	$[1,0] \to \textit{Fail}$	$[1,0] \to S_4$
$[0,1] \to S_1$	$[0,1] \to \textit{Fail}$	$[0,1] \to S_5$	$[0,1] \to S_5$
$[1,1] \to \textit{Fail}$	$[1,1] \to S_2$	$[1,1] \to S_6$	$[1,1] \to S_6$
State S_5	State S_6	State S_7	State S_8
$[0,0] \to S_5$	$[0,0] \to S_4$	$[0,0] \to \textit{Fail}$	$[0,0] \to S_4$
$[1,0] \to S_3$	$[1,0] \to S_6$	$[1,0] \to S_3$	$[1,0] \to \textit{Fail}$
$[0,1] \to \textit{Fail}$	$[0,1] \to S_2$	$[0,1] \to S_7$	$[0,1] \to S_2$
$[1,1] \to S_1$	$[1,1] \to \textit{Fail}$	$[1,1] \to S_1$	$[1,1] \to S_8$

Table 5.2. Rule Table for Fractal 781

For example, when in state S_4, if the next bit-pair is $[0, 1]$, then Table 5.1 indicates that the resulting state should be the state corresponding to $M_4M_Q = M_4M_8 = M_5$ or S_5. Thus, our table tells us that when in state S_4, if we are "processing" the bit-pair $[0, 1]$, the new state is S_5.

For the point $\left(\frac{2}{5}, \frac{1}{3}\right)$ we have

$$\begin{aligned}(x, y) &= \left(\frac{2}{5}, \frac{1}{3}\right)\\ &= (.011001100110\ldots, .0101010\ldots)\\ &= (.x_1x_2x_3\ldots, .y_1y_2y_3\ldots)\end{aligned}$$

We begin in state S_1 and process the successive bit-pairs leading to the following sequence of states:

$$\begin{aligned} S_1 &\to r(S_1, [x_1, y_1]) = r(S_1, [0,0]) = S_7 \\ &\to r(S_7, [x_2, y_2]) = r(S_7, [1,1]) = S_1 \\ &\to r(S_1, [x_3, y_3]) = r(S_1, [1,0]) = S_8 \\ &\to r(S_8, [x_4, y_4]) = r(S_8, [0,1]) = S_2 \\ &\to r(S_2, [x_5, y_5]) = r(S_2, [0,0]) = S_7 \\ &\to r(S_7, [x_6, y_6]) = r(S_7, [1,1]) = S_1 \\ &\to r(S_1, [x_7, y_7]) = r(S_1, [1,0]) = S_8 \\ &\to \ldots \end{aligned}$$

We can see that this pattern will repeat itself and thus this sequence will never reach the fail state. Hence, $\left(\frac{2}{5}, \frac{1}{3}\right)$ is in the fractal. However,

$$\begin{aligned} (x, y) &= \left(\frac{2}{5}, \frac{3}{8}\right) \\ &= (.01100110\ldots, .0110000\ldots) \end{aligned}$$

leads to the following sequence of states:

$$\begin{aligned} S_1 &\to r(S_1, [0,0]) = S_7 \\ &\to r(S_7, [1,1]) = S_1 \\ &\to r(S_1, [1,1]) = Fail \end{aligned}$$

Since a similar result occurs if $\frac{3}{8}$ is written as $.0101111\ldots$, the point $\left(\frac{2}{5}, \frac{3}{8}\right)$ is not in the fractal.

We will show that the set of points that have a binary representation such that processing the bit-pairs never reaches the *Fail* state is precisely the set of points in the fractal. However, we note that while this is a well-defined set, the decision of whether or not a specific point is in the set or not may not be decidable. For rational numbers, we will always either reach the *Fail* state or land in a cycle that does not fail, so we will be able to decide if the point is or is not in the fractal. Unfortunately, for irrational numbers, unless the *Fail* state is reached, we will be unable to decide.

Before looking at why our algorithm works, we explore an alternative approach for visualizing the rule table. Because the algorithm proceeds from state to state processing bit-pairs, a natural model to use is that of finite automata. (See [6], for example, for background on finite automata.) For any fractal PQR we construct a directed graph where each vertex on the graph corresponds to one of our eight possible states. Each state has three edges leaving it corresponding to the three possible next states, other than *Fail*, as given by the rule table. Each edge is labeled with the bit-pair that leads to that next state. The input string for the finite automata is the sequence of bit-pairs arising from the binary representation of the point (x, y) that we are testing. We begin in state S_1 and process the successive bit-pairs of (x, y) moving from state to state along the edges specified by the bit-pairs. In finite automata, only allowable symbols appear on edges and so bit-pairs that lead to the *Fail* state are not present. As long as we never reach the *Fail* state, we can continue moving from vertex to vertex in the graph processing the bit-pairs. If a point (x, y) is not in the fractal PQR, then at some step in our processing we will be in a state and looking at a bit-pair that is not one of the labels on any of the three edges leaving that state. Thus, all strings of bit-pairs that can be processed indefinitely by our graph are precisely the strings of bit-pairs corresponding to points in our fractal.

We illustrate this graphical representation of the rule table for fractal 781 in Figure 5.3. We start in state S_1. There are three edges leaving S_1 labeled $[0,0]$, $[0,1]$ and $[1,0]$. Corresponding to Table 5.2, the edge labeled with bit-pair $[0,0]$ leads to state S_7, the edge labeled with bit-pair $[0,1]$ leads back to state S_1 and the edge labeled with $[1,0]$ leads to state S_8. The bit-pair $[1,1]$ does not appear on an edge leaving S_1, since $[1,1]$ leads to the *Fail* state. To process a point (x, y), we find its binary notation $(.x_1x_2x_3\ldots, .y_1y_2y_3\ldots)$. Then starting in state S_1, we move along the edge corresponding to the bit pair $[x_1, y_1]$ to find the next state. Once there, we use $[x_2, y_2]$ to select the edge to follow to find the third state, and so on.

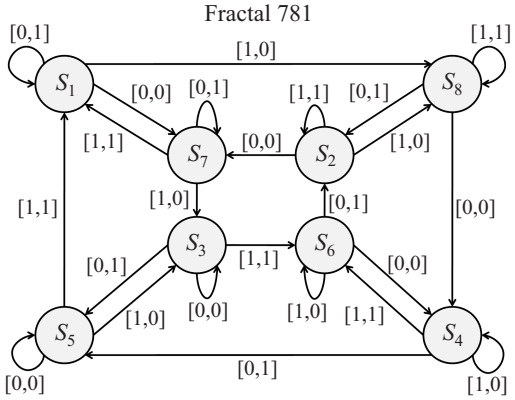


Figure 5.3. The graph of the Rule Table

We now explore the relationship between the rule table of a fractal (whether in table or graphical form) and the picture of the fractal. Each of the eight states $S_1, S_2, \ldots, S_8$ has a direct relationship to each of the corresponding matrices $M_1, M_2, \ldots, M_8$. As we process the bit-pairs of a point (x, y), we can think of this as zooming in on the point (x, y) by looking at successively smaller squares, in particular, the squares of side lengths $\left(\frac{1}{2}\right)^k$ that have lower left corners at $(.x_1x_2x_3\ldots x_k, .y_1y_2y_3\ldots y_k)$. If we find ourselves in state S_j at the kth step, the fractal that we see in this square is the fractal transformed by M_j. For example, consider again processing $\left(\frac{2}{5}, \frac{1}{3}\right) = (.011001100110\ldots, .0101010\ldots)$ for fractal 781. Figures 5.4a–5.4d show the successive collections of squares we are looking at. We start in state S_1 and M_1 is the identity matrix. We have processed 0 bit-pairs; thus, if we look at the unit square—the square of side length $\left(\frac{1}{2}\right)^0$ with lower left corner at the origin—we see the original fractal. After processing the first bit-pair, we find ourselves in state S_7. Now M_7 is the matrix corresponding to rotation by $\frac{\pi}{2}$, followed by a reflection across the x-axis. We have processed 1 bit-pair, thus, if we look at the square of side length $\frac{1}{2}$ with lower left corner $(.x_1, .y_1) = (0, 0)$,

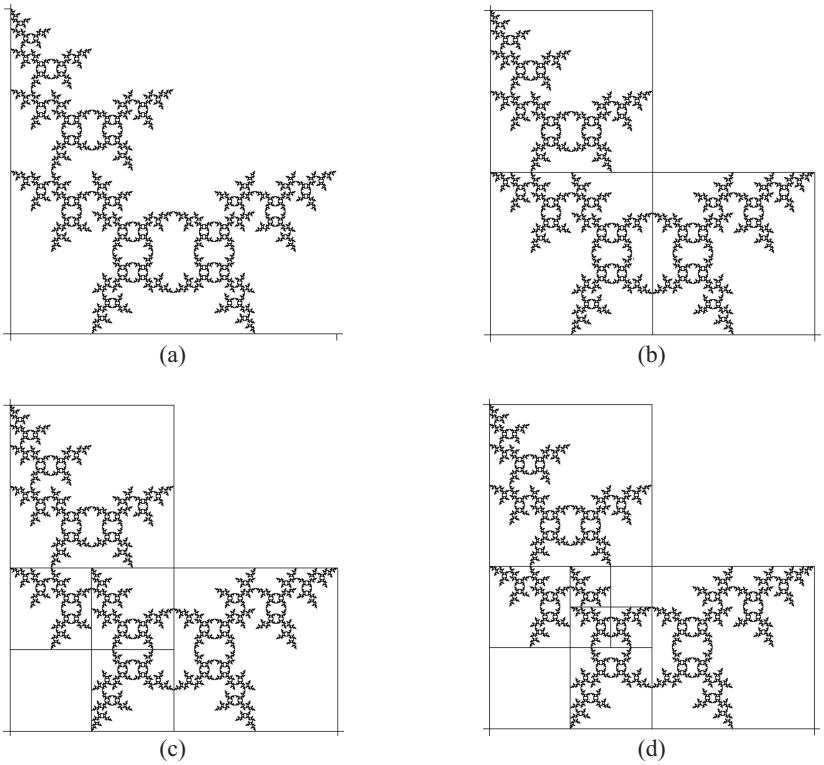


Figure 5.4.

we will see the fractal scaled by $\frac{1}{2}$ and transformed by a rotation of $\frac{\pi}{2}$, followed by a reflection across the x-axis. The next bit-pair is $[1, 1]$: we are currently in S_7 and so following the edge labeled $[1, 1]$ out of S_7, or using the rule table, we land in state S_1. We have processed two bit-pairs and if we look at the square of side length $\left(\frac{1}{2}\right)^2 = \frac{1}{4}$ with lower left corner $(.x_1x_2, .y_1y_2) = (.01, .01) = \left(\frac{1}{4}, \frac{1}{4}\right)$ we have the fractal scaled by $\frac{1}{4}$ and transformed by M_1. Since M_1 is the identity transformation, we see the fractal in its original orientation. The next bit-pair $[1, 0]$ leads to state S_8 and in the square of side length $\frac{1}{8}$ with lower left corner $(.x_1x_2x_3, .y_1y_2y_3) = (.011, .010) = \left(\frac{3}{8}, \frac{1}{4}\right)$ we see the fractal scaled by $\frac{1}{8}$ and transformed by M_8, namely a rotation of $\frac{\pi}{2}$ followed by reflections across both axes.

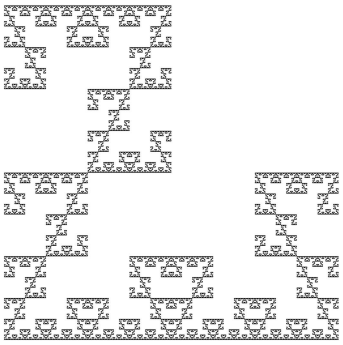
In general, we can use a graph like that in Figure 5.3, to track what any Sierpiński-like fractal will look like in any square of side length $\frac{1}{2^k}$ with lower left corner

$$(.x_1x_2x_3 \ldots x_k, .y_1y_2y_3 \ldots y_k) = \left(\sum_{i=1}^{k} \frac{x_i}{2^i}, \sum_{i=1}^{k} \frac{y_i}{2^i}\right).$$

By following the directed edges corresponding to the bit-pairs $[x_i, y_i]$ for k steps, we will find ourselves in some state S_j; the corresponding matrix M_j will tell us how the fractal appears—what transformation to apply to the original fractal—in that small square. Squares in the picture of the fractal that are blank, i.e. have no points, correspond to having landed in the *Fail* state when using the rule table. These are also squares that have coordinates in the lower left corner whose bit-pairs cannot be processed in the directed graph; at some step we will be in a state where the needed bit-pair is not one of the three allowed.

Interestingly, the graphical version of the rule table for fractal 781 bears a striking similarity to a Cayley diagram of a group and we note that this is in fact exactly what we have reproduced, although using an unusual set of generators. The underlying group is the group of transformations $M_1, M_2, \ldots, M_8$, or one of its subgroups, which is just the group of symmetries of a square, that is, the dihedral group of order 8 or D_4. However, instead of using the two standard generators, M_5 and M_3 in our notation, we are using M_7, M_8 and M_1 as the generators. The use of M_1, the identity element, as a "generator" accounts for the self-loops at each vertex. See [4] or [3] for example, for background on Cayley diagrams. In essence, the Cayley diagram of a group is a directed graph with the elements as vertices and the edges encoding the results of the group operation. The edges leaving S_1 in the graph of our rule table for fractal 781 are labeled $[0, 0]$, $[0, 1]$ and $[1, 0]$. Referring back to Table 5.1, we see that these bit-pairs can be uniquely associated with M_P, M_Q and M_R which in this case are M_7, M_8 and M_1. The association is just which matrix is multiplied by M_1 to determine the new state. In general, in Table 5.1 we see that when in state S_j the new state is found by multiplying M_jM_P, M_jM_Q or M_jM_R. The choice of M_P, M_Q or M_R is determined by which bit-pair is being processed. Thus, the edges leaving S_2 are labeled $[0, 0]$, $[1, 0]$ and $[1, 1]$ and these, in order, are associated with M_Q, M_P and M_R in Table 5.1, or for fractal 781 specifically, with M_8, M_7 and M_1. Thus, for each edge in the graph of the rule table, we can replace the bit-pair that appears on the edge with one of M_7, M_8 or M_1. Doing so gives us the Cayley graph for D_4 using the generators M_7, M_8 or M_1. If M_i appears on the directed edge from S_j to S_k, then $M_jM_i = M_k$. We note that our sets of generators will be unusual: they will not be minimal and will frequently contain the same generator more than once.

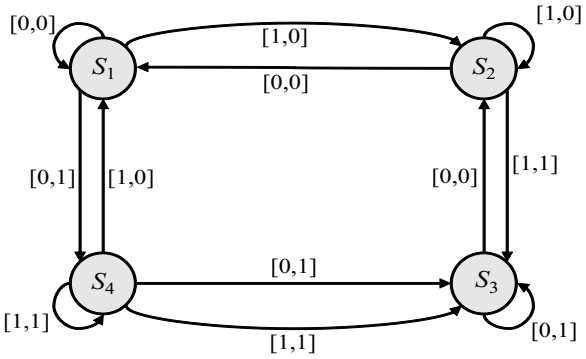
For an arbitrary fractal PQR, the graph of the rule table can take one of several forms. For the Sierpiński triangle itself, or fractal 111, the graph consists of just a single state S_1 with three self-edges corresponding to the three allowed bit-pairs $[0, 0]$, $[1, 0]$, and $[0, 1]$. Thus, the group corresponding to the Sierpiński triangle is just the subgroup with one element, the identity, and its Cayley diagram consists of just the single element. For fractals such as 121 or 441, only two states are needed. Thus, fractal 121 includes only S_1 and S_2, a reflection across the y-axis. M_2 is of order two; hence, the graph of the rule table for fractal 121 and the Cayley diagram for the subgroup $\{M_1, M_2\}$ each have two vertices. Similarly, fractal 441 uses only the transformation M_4, which is a reflection across both the x and y-axes (the same as a rotation of π), which is another element of order two in D_4. Fractals that use reflections across each of the axes but no rotations will have graphs of their rule tables that use states S_1, S_2, S_3 and S_4 and fractals that use rotations of $\frac{\pi}{2}$, π and $\frac{3\pi}{2}$ will have graphs of their rule tables that use states S_1, S_5, S_4 and S_8. These are the graphs corresponding to the Cayley diagrams of the two subgroups of order four of D_4. For fractals with both rotations and reflections, all eight states are needed and the graphs of their rule tables correspond to the Cayley diagram for the full group D_4.



(a) Fractal 124

State S_1	State S_2	State S_3	State S_4
$[0,0] \to S_1$	$[0,0] \to S_1$	$[0,0] \to S_2$	$[0,0] \to \mathit{Fail}$
$[1,0] \to S_2$	$[1,0] \to S_2$	$[1,0] \to \mathit{Fail}$	$[1,0] \to S_1$
$[0,1] \to S_4$	$[0,1] \to \mathit{Fail}$	$[0,1] \to S_3$	$[0,1] \to S_3$
$[1,1] \to \mathit{Fail}$	$[1,1] \to S_3$	$[1,1] \to S_4$	$[1,1] \to S_4$

(b) Rule Table for Fractal 124



(c) Rule Table Graph for Fractal 124

Figure 5.5.

We illustrate these ideas with two more fractals, fractal 124 and fractal 468 (see Figures 5.5 and 5.6). Fractal 124 has the identity transformation in the bottom left quadrant, a reflection across the y-axis in the bottom right quadrant and a reflection across both axes in the top left quadrant. The graph of its rule table consists of just four states.

For fractal 468 (Figure 5.6) we have modified the graph of the rule table. We have left off the bit-pairs, collapsed pairs of edges that go between two states—one in each direction—to a single edge and have coded the edges by which transformation was used in the rule table to determine the new state. Thus, for each element M_i in the group, the results of multiplying on the right by each of M_2, M_4 and M_6 can be found by following the appropriate edge. The states have been labeled with just the subscript.

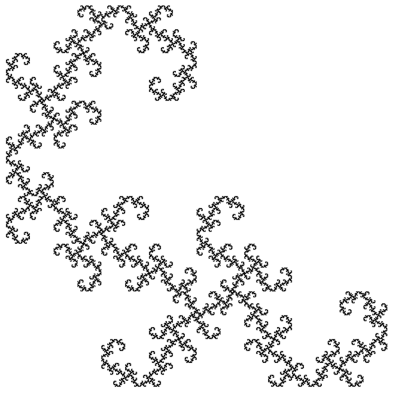
We now turn to the question of why the "rule" table is constructed the way it is and outline a proof that the set of points that never reach the *Fail* state when processed by our algorithm is in fact the fractal PQR. We begin by looking carefully at how the functions f_0, f_1 and f_2 alter the bit-pairs of a point (x, y). Each function has the form

$$f_i\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{2}M_j\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_{x_i} \\ t_{y_i} \end{pmatrix};$$

multiplication by 1/2 will shift all the bits to the right by one place, multiplication by M_j may multiply the bits of one or both of x and y by -1 and may switch the bits of x with those of y if there is a rotation, and the translation will be an addition of $1/2$ or 1 depending on the combination of i and j. For example, for our fractal 781, we have that f_0 uses transformation M_7 giving

$$f_0\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$$

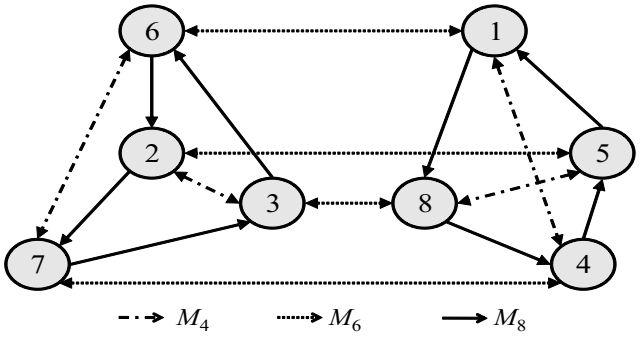
where the translation $(1/2, 1/2)$ has been determined so that $f_0(Q) \subseteq Q_0$. Looking at the binary expansion of this



(a) Fractal 468

State S_1	State S_2	State S_3	State S_4
$[0,0] \to S_4$	$[0,0] \to S_5$	$[0,0] \to S_6$	$[0,0] \to Fail$
$[1,0] \to S_6$	$[1,0] \to S_3$	$[1,0] \to Fail$	$[1,0] \to S_5$
$[0,1] \to S_8$	$[0,1] \to Fail$	$[0,1] \to S_2$	$[0,1] \to S_7$
$[1,1] \to Fail$	$[1,1] \to S_7$	$[1,1] \to S_8$	$[1,1] \to S_1$
State S_5	State S_6	State S_7	State S_8
$[0,0] \to S_1$	$[0,0] \to S_7$	$[0,0] \to Fail$	$[0,0] \to S_3$
$[1,0] \to S_8$	$[1,0] \to S_2$	$[1,0] \to S_4$	$[1,0] \to Fail$
$[0,1] \to Fail$	$[0,1] \to S_1$	$[0,1] \to S_3$	$[0,1] \to S_5$
$[1,1] \to S_2$	$[1,1] \to Fail$	$[1,1] \to S_6$	$[1,1] \to S_4$

(b) Rule Table for Fractal 468



(c) Rule Table Graph for Fractal 468

Figure 5.6.

point, we see that

$$f_0\begin{pmatrix} .x_1x_2x_3\ldots \\ .y_1y_2y_3\ldots \end{pmatrix} = \begin{pmatrix} -.0y_1y_2y_3\ldots \\ -.0x_1x_2x_3\ldots \end{pmatrix} + \begin{pmatrix} .01111\ldots \\ .01111\ldots \end{pmatrix}$$
$$= \begin{pmatrix} .0\overline{y}_1\overline{y}_2\overline{y}_3\ldots \\ .0\overline{x}_1\overline{x}_2\overline{x}_3\ldots \end{pmatrix}$$

where we have chosen to write $1/2$ as $.01111\ldots$ and we use the notation $\overline{a}$ to indicate the binary complement of a, that is, $\overline{a} = 0$ if $a = 1$ and $\overline{a} = 1$ if $a = 0$. Now consider what happens if the point $(u, v) = f_0(x, y)$ is processed by our algorithm. Let $T_1, T_2, T_{3\ldots}$ be the sequence of states that are obtained when processing the point (x, y) and let $W_1, W_2, W_3\ldots$ be the states for (u, v). Since we always start in state S_1, we have that $T_1 = W_1 = S_1$. When processing (x, y) the first bit-pair is $[x_1, y_1]$ and $r(S_1, [x_1, y_1]) = T_2$. Since the first bit-pair of (u, v) is $[0, 0]$, we get that $W_2 = r(S_1, [0, 0]) = M_1M_P = S_7$. The next bit-pair for (u, v) is $[u_2, v_2] = [\overline{y}_1, \overline{x}_1]$ and we need to determine $r(S_7, [\overline{y}_1, \overline{x}_1])$. From the rule table in Table 1, we see that if T_2 is the state obtained from $r(S_1, [x_1, y_1])$, then the state for $r(S_7, [\overline{y}_1, \overline{x}_1])$ will just be M_7T_2. Furthermore, this holds in general: if $r(T_k, [x_k, y_k]) = T_{k+1}$, then $W_{k+1} = r(W_k, [u_k, v_k]) = r(W_k, [\overline{y}_{k-1}, \overline{x}_{k-1}]) = M_P T_k$. These observations lead us to the following.

Lemma 5.1. *Suppose $\{T_k\}$ is the sequence of states obtained from processing a point (x, y) with our algorithm. If $\{A_k\}$, $\{B_k\}$, and $\{C_k\}$ are the sequence of states obtained when processing the points $f_0(x, y)$, $f_1(x, y)$ and $f_2(x, y)$ respectively in fractal PQR, then for $k = 1, 2, 3, \ldots$, we have $A_{k+1} = M_P T_k$, $B_{k+1} = M_Q T_k$ and $C_{k+1} = M_R T_k$.*

Proof. The details of this proof follow from a careful examination of the possible functions f_i and the rule table given in Table 5.1. □

We also have the reverse. Consider fractal 781 and suppose we have a point (x, y) and suppose furthermore that $[x_1, y_1] = [0, 0]$. This means that we can construct a point that will be mapped to (x, y) by reversing the action of f_0

on the bit-pairs. (If $[x_1, y_1]$ had been $[1, 0]$ we would use f_1 and if $[x_1, y_1]$ had been $[0, 1]$ we would use f_2.) That is, if we start with the point

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} .\overline{y}_2\overline{y}_3\overline{y}_4\cdots \\ .\overline{x}_2\overline{x}_3\overline{x}_4\ldots \end{pmatrix},$$

we know that we will have

$$f_0\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} .0x_2x_3x_4\ldots \\ .0y_2y_3y_4\ldots \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}.$$

Furthermore, from Lemma 5.1, we know how the sequence of states for the point (x, y) and (a, b) must be related. Namely, if the sequence for (x, y) is $\{T_k\}$, then the sequence for (a, b) must be S_1, $M_7^{-1}T_3$, $M_7^{-1}T_4, \ldots$ or more generally $\left\{M_7^{-1}T_{k+1}\right\}$. Again, this holds in general: if the point (x, y) starts with $[0, 0]$ there will be a point with sequence $\left\{M_P^{-1}T_{k+1}\right\}$, if it starts with $[1, 0]$ there will be a point with sequence $\left\{M_Q^{-1}T_{k+1}\right\}$ and if it starts with $[1, 0]$ there will be a point with sequence $\left\{M_R^{-1}T_{k+1}\right\}$. In each case, the point is found by removing the first bit-pair and reversing the action of f_0, f_1 or f_2 respectively on the remaining bit-pairs. This gives us our second lemma.

Lemma 5.2. *Suppose $\{T_k\}$ is the sequence of states obtained from processing a point (x, y) with our algorithm. Then there is a point (a, b), such that for one of $i = 0$, 1 or 2, we have $f_i(a, b) = (x, y)$ and the sequence of states when processing (a, b) will be $\left\{M_P^{-1}T_{k+1}\right\}$ if $i = 0$, $\left\{M_Q^{-1}T_{k+1}\right\}$ if $i = 1$ and $\left\{M_R^{-1}T_{k+1}\right\}$ if $i = 2$.*

Proof. Again, the details of this proof follow the above discussion and looking at the possible functions f_i. □

Our final lemma will be about the topology of the set of points that never reach the *Fail* state. We claim that this set contains all its limit points.

Lemma 5.3. *Let B be the set of points in the unit square that have a binary representation such that, when processed by our algorithm, the Fail state is never reached. Then B contains all its limit points.*

Proof. Suppose not, that is, suppose there is a sequence of points $\{p_n\}$ in the unit square such that processing each point $p_n = (x^n, y^n)$ with our algorithm leads to a sequence of states $\left\{T_k^n\right\}$ where each T_k^n is not the *Fail* state. Further, suppose that, using the normal Euclidean metric, $p_n \to p$ where $p = (x, y)$, and suppose also that the sequence of states obtained when processing p does reach the *Fail* state and that any other binary representation of p also fails. Denote the bit-pairs of p by $[x_k, y_k]$ and those of each p_n by $\left[x_k^n, y_k^n\right]$. Let T_{K+1} be the first state that fails when processing p, so that the Kth bit-pair of p is the one that leads to the *Fail* state. If there is more than one binary representation, choose the one that fails the latest. Next, determine the position of the next bit in x that is the same as x_K and the next bit in y that is the same as y_K and call these positions m_x and m_y. We know that these positions exist since otherwise, x or y would end in all $0's$ or all $1's$ and in this case we could use a different binary representation that would fail later. Let $m = \max\left\{m_x, m_y\right\}$. We will show that for all n,

$$|p_n - p| > \frac{1}{2^m}$$

contradicting the fact that $p_n \to p$.

Consider any p_n. Since each p_n does not fail and p does fail, we know that the bit-pairs of p_n and p differ no later than in the Kth position. Since at least one of the bits in the bit-pair must be different, suppose without loss of generality, that the binary representations of x^n and x differ. Suppose further that they differ first at the Kth position, since this is the latest that the bits may differ. Then

$$\begin{aligned} |p_n - p| &= \sqrt{(x^n - x)^2 + (y^n - y)^2} \\ &\geq |x^n - x| \\ &= \frac{1}{2^K}\left|\left(x_K^n - x_K\right) + \frac{1}{2}\left(x_{K+1}^n - x_{K+1}\right) + \frac{1}{4}\left(x_{K+2}^n - x_{K+2}\right)\ldots\right| \end{aligned}$$

Now we know that $\left|x_K^n - x_K\right| = 1$ and we also know that $\left(x_{m_x}^n - x_{m_x}\right)$ is either 0 or has the same sign as $\left(x_K^n - x_K\right)$ since the bit x_{m_x} is equal to the bit x_K. Thus,

$$\begin{aligned}
|p_n - p| &= \frac{1}{2^K}\left|\left(x_K^n - x_K\right) + \frac{1}{2}\left(x_{K+1}^n - x_{K+1}\right) + \frac{1}{4}\left(x_{K+2}^n - x_{K+2}\right)\ldots\right| \\
&\geq \frac{1}{2^K}\left(1 - \frac{1}{2} - \frac{1}{4} - \ldots - \frac{1}{2^{m_x-K-1}} - \frac{1}{2^{m_x-K+1}} - \frac{1}{2^{m_x-K+2}} - \ldots\right) \\
&= \frac{1}{2^K}\left(\frac{1}{2^{m_x-K-1}} - \frac{1}{2^{m_x-K}}\right) \\
&= \frac{1}{2^{m_x-1}} - \frac{1}{2^{m_x}} \\
&\geq \frac{1}{2^m}.
\end{aligned}$$

Similar arguments can be made when the bit-pairs of p_n and p differ before the Kth position or differ only in the bit strings of the y coordinates. Since each p_n and p are therefore a fixed distance apart, we have contradicted that $p_n \to p$. Hence, if each p_n does not fail, it must be the case that p does not fail and hence, B contains all its limit points. □

We are now ready to state our main result.

Theorem 5.1. *Let B be the set of points in the unit square that never reach the fail state when processed by our algorithm for fractal PQR and let A be the fractal given by the IFS with transformations M_P, M_Q and M_R. Then $B = A$.*

Proof. We will show that $B = A$ by showing that the set of points B has the property that $f_0(B) \cup f_1(B) \cup f_2(B) = B$ and that the set B is compact. Since A is the unique compact set with this property, then $B = A$.

We begin by showing that $f_0(B) \cup f_1(B) \cup f_2(B) \subseteq B$. So suppose that (x, y) is a point such that the algorithm never reaches the *Fail* state. We need to show that the point $f_i(x, y)$ never reaches the *Fail* state for $i = 0, 1, 2$. Let $(u, v) = f_0(x, y)$. When processing the bit-pairs of (x, y), the algorithm produces a sequence of states, say $T_1, T_2, \ldots$. But from Lemma 5.1, we know that when processing the bit-pairs of (u, v), the algorithm produces the sequence of states $S_1, M_P T_1, M_P T_2, \ldots$ and thus will also never reach the fail state. Hence $(u, v) \in B$.

Next we show that $B \subseteq f_0(B) \cup f_1(B) \cup f_2(B)$. Let $(x, y) \in B$, i.e., let (x, y) be a point such that the algorithm never reaches the fail state and suppose the algorithm produces the sequence of states $T_1, T_2, \ldots$. From Lemma 2, we know that if $[x_1, y_1] = [0, 0]$ then there exists a point (a, b) such that $f_0(a, b) = (x, y)$ and (a, b) has the sequence of states $\left\{M_P^{-1} T_{k+1}\right\}$, that if $[x_1, y_1] = [1, 0]$ then there exists a point (a, b) such that $f_1(a, b) = (x, y)$ and (a, b) has the sequence of states $\left\{M_Q^{-1} T_{k+1}\right\}$, and that if $[x_1, y_1] = [0, 1]$ then there exists a point (a, b) such that $f_2(a, b) = (x, y)$ and (a, b) has the sequence of states or $\left\{M_R^{-1} T_{k+1}\right\}$. In each case, if $\{T_k\}$ never reaches the *Fail* state, then neither will the other sequence. Hence $(a, b) \in B$.

Thus we have that our set B is fixed by the IFS.

Finally, by Lemma 3 we know that B contains all its limit points and, thus, is closed. Since B is contained in the unit square, it is bounded and hence compact. □

Bibliography

[1] Barnsley, M., *Fractals Everywhere*, 2nd edition, Academic Press Professional, 1993.

[2] Devaney, R., *A First Course in Chaotic Dynamical Systems*, Addison-Wesley, 1992.

[3] Gallian, J., *Contemporary Abstract Algebra*, 7th edition, Brooks Cole, 2009.

[4] Grossman I. and W. Magnus, *Groups and their Graphs*, Random House, 1964.

[5] Gulick, D., *Encounters with Chaos*, McGraw-Hill, 1992.

[6] Linz, P., *An Introduction to Formal Languages and Automata*, 3rd edition, Jones and Bartlett, 2001.

[7] Parris, R., math.exeter.edu/rparris/winfeed.html, 2008.

[8] Peitgen, H.O., H. Jurgens and D. Saupe, *Chaos and Fractals: New Frontiers of Science*, Springer-Verlag, 1992.

[9] Riddle, L., *Classic Iterated Function Systems*, ecademy.agnesscott.edu/~lriddle/ifs/ifs.htm, 2008.

[10] ——, *IFS Construction Kit*, ecademy.agnesscott.edu/~lriddle/ifskit/index.htm, 2008.

6

Fractals in the 3-Body Problem Via Symplectic Integration

Daniel Hemberger
Cornell University

James A. Walsh
Oberlin College

6.1 Introduction

The statement of the n-body problem is tantalizingly simple: Given the present positions and velocities of n celestial bodies, predict their motions under Newton's inverse square law of gravitation for all future time and deduce them for all past time.

This simplicity belies the fact that efforts to solve this problem, beginning particularly with the work of Henri Poincaré in the late 19th century, essentially led to the creation of the field of dynamical systems. The study of the n-body problem remains an active area of research (see, for example, [17]).

It is intriguing to consider how Poincaré's research might have benefited had he access to modern computing capabilities. That he was able to discern what are now known as fractals and chaotic behavior in the 3-body problem [19] using only paper, pencil, and mental acumen (ferocious though it was) is remarkable.

It is safe to assume, nonetheless, that even Poincaré would have gained further insight into fractal geometry and chaotic dynamics had he access to modern numerical integration algorithms. In particular, given that the equations of motion for the 3-body problem are *Hamiltonian* in nature, it is now clear that he would have used a *symplectic integration algorithm* to numerically compute solutions.

The development of new numerical integration algorithms for Hamiltonian systems of ODEs and, more generally, for systems of equations admitting a geometric structure, began in the 1990s and continues to this day [6]. These algorithms have proven far superior to Runge-Kutta type algorithms in approximating the long-term, global, qualitative behavior of solutions. It is fun to imagine Poincaré staring at a phase space on a computer screen, as might a modern dynamicist, searching for insights which could lead to another new theorem.

In this paper we investigate fractals arising in the 3-body problem. We do this via the simplified, low-dimensional variation of this problem known as the trilinear 3-body problem [15]. Along the way we define what Hamiltonian systems of ODEs and symplectic maps are. We also present an example of a symplectic integration algorithm (SIA) and illustrate why an SIA is ideally suited for generating the fractal structure which can arise in phase space for Hamiltonian systems of equations.

Many approaches to designing SIAs involve the use of generating functions or Lie algebraic techniques ([5], [10]). Our presentation of this type of algorithm is more elementary and well-suited for undergraduates. This article is in fact an extension of a research project undertaken by the first author in his junior year while a member of an introductory dynamical systems course taught by the second author. Hemberger's SIA code was written and executed in *Matlab*.

We conclude this paper with a comparison of the performance of this SIA vis-à-vis Runge-Kutta methods [11, Ch. 2]. The superiority of the symplectic algorithm is striking (see Figures 6.9, 6.10, and 6.11, for example).

6.2 Hamiltonian Systems

Hamiltonian systems of equations often arise when modeling physical systems in which there is a conserved quantity such as energy. Their importance in the study of ODEs at the undergraduate level is indicated by their inclusion in standard sophomore-level ODE texts such as [3] and [4].

We begin with the definition of a Hamiltonian system, to be followed by two well-known illustrative examples and the introduction of the trilinear 3-body model Hamiltonian. Let $H : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a C^2 function, where points in $\mathbb{R}^d \times \mathbb{R}^d$ take the form $(\mathbf{q}, \mathbf{p}) = (q_1, \ldots, q_d, p_1, \ldots, p_d)$. The Hamiltonian system of ODEs with Hamiltonian H is given by the system of $2d$ first-order equations

$$\dot{q}_i = \frac{\partial H}{\partial p_i}, \qquad \dot{p}_i = -\frac{\partial H}{\partial q_i}, \quad i = 1, \ldots, d, \tag{6.1}$$

with dots representing differentiation with respect to time. The integer d is the *number of degrees of freedom*, while $\mathbb{R}^d \times \mathbb{R}^d$ is the *phase space*. In applications to mechanics, the $\mathbf{q}$ variables are often generalized coordinates, the $\mathbf{p}$ variables are conjugate generalized momenta, and H represents the total mechanical energy.

A Hamiltonian function H is said to represent a *conserved quantity* for system (6.1) for the following reason. Suppose $(\mathbf{q}(t), \mathbf{p}(t))$ is a solution to (6.1). By the multivariable chain rule and equation (6.1) we then have

$$\frac{d}{dt} H(\mathbf{q}(t), \mathbf{p}(t)) = \sum_{i=1}^{d} \frac{\partial H}{\partial q_i}\frac{dq_i}{dt} + \sum_{i=1}^{d} \frac{\partial H}{\partial p_i}\frac{dp_i}{dt} = \sum_{i=1}^{d} \left(-\frac{dp_i}{dt}\right)\frac{dq_i}{dt} + \sum_{i=1}^{d} \frac{dq_i}{dt}\frac{dp_i}{dt} = 0.$$

Hence the function H is constant along solution curves of system (6.1) or, put another way, solutions to (6.1) lie on level sets of H. This reduces the dimension of the problem by one, allowing one to sketch solution curves in phase space by plotting level sets of H, without recourse to actual formulas for $\mathbf{q}(t)$ and $\mathbf{p}(t)$ (see Figures 6.1 and 6.2).

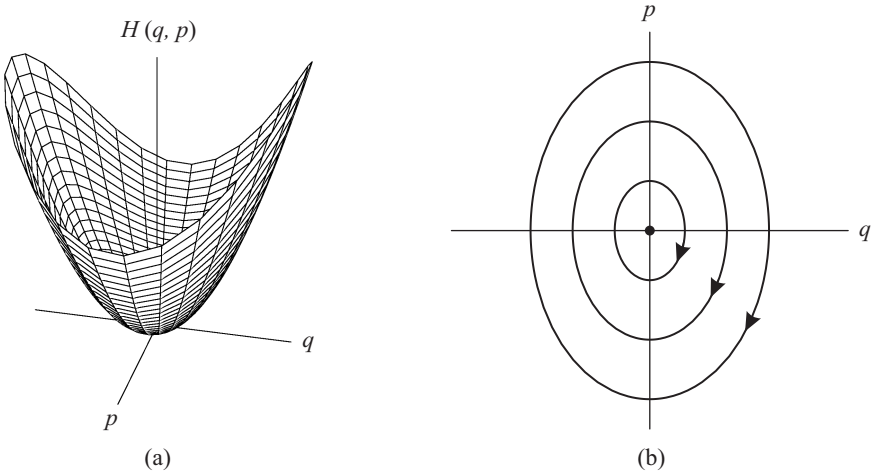


Figure 6.1. (a) The Hamiltonian $H(q, p) = \frac{1}{2m}p^2 + \frac{k}{2}q^2$. (b) The (q, p)-phase plane for the harmonic oscillator ($m = 1, k = 2$). Level sets of the Hamiltonian provide solution curves in the (q, p)-phase plane.

Our first example of a Hamiltonian system is one encountered by all students of introductory differential equations, namely that derived from the undamped, unforced harmonic oscillator:

$$m\ddot{q} + kq = 0, \qquad m, k > 0. \tag{6.2}$$

In equation (6.2), $q = q(t)$ is the position of a block of mass m attached to a spring with spring constant k, relative to equilibrium at time t, while two dots represents the second derivative with respect to t.

As Hamiltonian systems are first-order systems of ODEs, we introduce a second variable $p = m\dot{q}$ (the conjugate momentum), so that $\dot{p} = m\ddot{q} = -kq$, converting (6.2) into the first-order system

$$\begin{aligned} \dot{q} &= \frac{1}{m}p \\ \dot{p} &= -kq. \end{aligned} \tag{6.3}$$

Solutions to (6.3) are curves in the (q, p)-phase plane (Figure 6.1(b)); one can then recover the time evolution of the position of the block by projecting onto the q-coordinate while moving along a solution curve.

Note that the scalar function $H : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, $H(q, p) = \frac{1}{2m}p^2 + \frac{k}{2}q^2$ satisfies

$$\begin{aligned} \dot{q} &= \frac{\partial H}{\partial p} \\ \dot{p} &= -\frac{\partial H}{\partial q}, \end{aligned} \tag{6.4}$$

that is, H satisfies equation (6.1) with $d = 1$. This implies then that H is a Hamiltonian function for system (6.3), so that level sets of H provide solution curves in the $\mathbb{R} \times \mathbb{R}$ phase space (Figure 6.1).

Also of note is the fact H separates into the sum of $A(p) = \frac{1}{2m}p^2$ and $U(q) = \frac{k}{2}q^2$; the former function can be interpreted as the kinetic energy, while the latter is the potential energy. Total energy is thus conserved along solution curves of system (6.3). Were we to add damping or external forcing to the model, this Hamiltonian structure would be lost as energy would no longer be conserved. For future reference, we note it is to separable Hamiltonians (of the form $H(\mathbf{q}, \mathbf{p}) = A(\mathbf{p}) + U(\mathbf{q})$) that the symplectic integration algorithm will be applied in section 6.6.

Our second example of a well-known Hamiltonian system is provided by the undamped, unforced ideal pendulum. In this model, $q = q(t)$ is the angle the rod forms with the vertical, g is the gravitational constant, and L is the length of the (massless) rod. The model ODE [3] is

$$mL\ddot{q} + mg \sin q = 0. \tag{6.5}$$

Again letting $p = m\dot{q}$, we convert (6.5) into the first-order system

$$\begin{cases} \dot{q} = \frac{1}{m}p \\ \dot{p} = -\frac{mg}{L}\sin q, \end{cases} \tag{6.6}$$

for which solution curves are sketched in Figure 6.2(b).

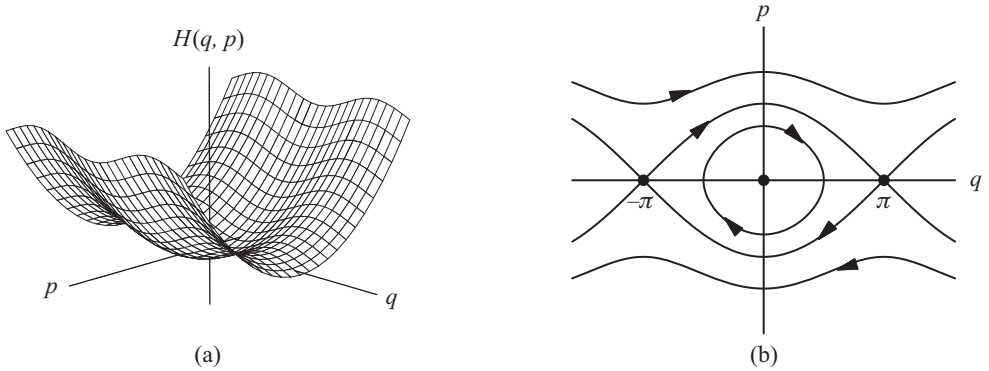


Figure 6.2. (a) The Hamiltonian $H(q, p) = \frac{1}{2m}p^2 - \frac{mg}{L}\cos q$. (b) The (q, p)-phase plane for the ideal pendulum ($m = 1, L = g$). Level sets of the Hamiltonian provide solution curves in the (q, p)-phase plane.

With neither damping nor forcing present, one might expect energy to be conserved for this model. This is indeed the case: the Hamiltonian function $H : \mathbb{R} \times \mathbb{R} \to \mathbb{R},\ H(q, p) = A(p) + U(q) = \frac{1}{2m}p^2 - \frac{mg}{L}\cos q$, where $A(p)$ and $U(q)$ represent the kinetic and potential energies, respectively, satisfies equation (6.4). Hence level curves of H provide solution curves to (6.6).

There is one degree of freedom in each of the harmonic oscillator and pendulum models. One implication of the Poincaré-Bendixson Theorem [14] is that chaotic behavior cannot occur in systems of first-order ODEs with phase space $\mathbb{R} \times \mathbb{R}$. As for the n-body problem, fractals and chaos do not occur when $n = 2$: J. Bernoulli proved in 1710 that the path followed by one body with respect to the other always lies along a conic section. At least 3 bodies are thus needed to potentially have fractal geometry and chaotic dynamics present in the n-body problem.

We now introduce a relatively simple variant of the 3-body problem and describe the inherent fractal structure Poincaré would have seen had he a modern workstation running a symplectic integration algorithm.

6.3 The Trilinear 3-Body Problem

The trilinear 3-body problem is arguably the simplest entrée into the study of the 3-body problem. The model yields a low-dimensional system which has the added benefit that no two bodies ever collide, ensuring that the corresponding vector field has no singularities.

Place bodies on each of three parallel lines in the plane as in Figure 6.3. The top and bottom lines are each separated from the center line by $\varepsilon/2$, for some fixed $\varepsilon > 0$. Let m_i and x_i denote the mass and (horizontal) position, measured from a chosen origin, of the ith body, $i = 1, 2, 3$. Let the bodies move without friction under Newton's inverse square law of gravitation. Given an initial state, what happens?

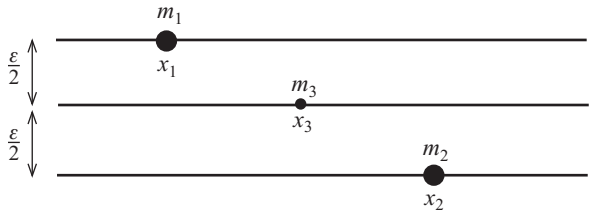


Figure 6.3. The trilinear 3-body model.

We begin by deriving the model differential equations. Though the equations may appear a bit unwieldy and yield an $\mathbb{R}^3 \times \mathbb{R}^3$ phase space, we will see that our study of model behavior can be reduced to two dimensions and iteration of a planar map.

Referring to Figure 6.4, we see that the distance from body 1 to body 2 is

$$D = \sqrt{(x_2 - x_1)^2 + \varepsilon^2}.$$

By Newton's inverse square law of gravitation, the component of the gravitational force exerted by m_2 on m_1 in the direction of motion of m_1 is

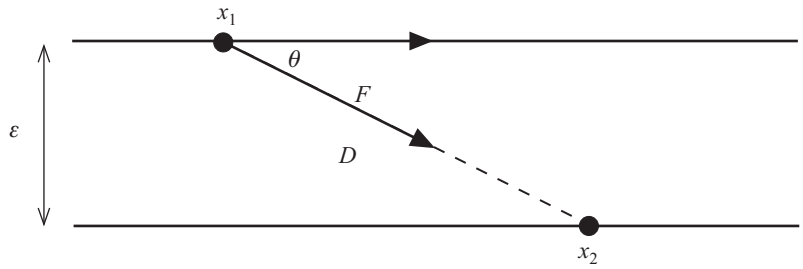
$$\|\mathbf{F}\| \cos\theta = \frac{m_1 m_2}{D^2} \cdot \frac{x_2 - x_1}{D} = \frac{m_1 m_2 (x_2 - x_1)}{((x_2 - x_1)^2 + \varepsilon^2)^{3/2}}.$$

We have set the gravitational constant G to equal 1 in the above, which can be accomplished by changing the unit of mass [16].

In a similar fashion, the component of the force m_3 exerts on m_1 in the direction of motion of m_1 is

$$\frac{m_1 m_3}{(\sqrt{(x_3 - x_1)^2 + (\varepsilon/2)^2})^2} \cdot \frac{x_3 - x_1}{\sqrt{(x_3 - x_1)^2 + (\varepsilon/2)^2}} = \frac{m_1 m_3 (x_3 - x_1)}{((x_3 - x_1)^2 + \varepsilon^2/4)^{3/2}}.$$

Computing the four remaining components of the forces one body exerts on another and using Newton's second law



$$D = \sqrt{(x_2 - x_1)^2 + \varepsilon^2}$$

Figure 6.4. Deriving the model equations.

yields the system of second-order equations

$$\begin{aligned}
m_1\ddot{x}_1 &= \frac{m_1m_2(x_2 - x_1)}{((x_2 - x_1)^2 + \varepsilon^2)^{3/2}} + \frac{m_1m_3(x_3 - x_1)}{((x_3 - x_1)^2 + \varepsilon^2/4)^{3/2}} \\
m_2\ddot{x}_2 &= \frac{m_1m_2(x_1 - x_2)}{((x_1 - x_2)^2 + \varepsilon^2)^{3/2}} + \frac{m_2m_3(x_3 - x_2)}{((x_3 - x_2)^2 + \varepsilon^2/4)^{3/2}} \\
m_3\ddot{x}_3 &= \frac{m_1m_3(x_1 - x_3)}{((x_1 - x_3)^2 + \varepsilon^2/4)^{3/2}} + \frac{m_2m_3(x_2 - x_3)}{((x_2 - x_3)^2 + \varepsilon^2/4)^{3/2}}.
\end{aligned} \tag{6.7}$$

Introducing the variables $q_i = x_i$ and $p_i = m_i\dot{q}_i$, $i = 1, 2, 3,$ converts (6.7) into the first-order system

$$\begin{aligned}
\dot{q}_i &= \frac{1}{m_i}p_i, \ i = 1, 2, 3, \quad \text{and} \\
\dot{p}_1 &= \frac{m_1m_2(q_2 - q_1)}{((q_2 - q_1)^2 + \varepsilon^2)^{3/2}} + \frac{m_1m_3(q_3 - q_1)}{((q_3 - q_1)^2 + \varepsilon^2/4)^{3/2}} \\
\dot{p}_2 &= \frac{m_1m_2(q_1 - q_2)}{((q_1 - q_2)^2 + \varepsilon^2)^{3/2}} + \frac{m_2m_3(q_3 - q_2)}{((q_3 - q_2)^2 + \varepsilon^2/4)^{3/2}} \\
\dot{p}_3 &= \frac{m_1m_3(q_1 - q_3)}{((q_1 - q_3)^2 + \varepsilon^2/4)^{3/2}} + \frac{m_2m_3(q_2 - q_3)}{((q_2 - q_3)^2 + \varepsilon^2/4)^{3/2}}.
\end{aligned} \tag{6.8}$$

Note that system (6.8) is defined for all $(\mathbf{q}, \mathbf{p})$ in $\mathbb{R}^3 \times \mathbb{R}^3$. Moreover, one can easily check that $H : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$,

$$\begin{aligned}
H(\mathbf{q}, \mathbf{p}) &= A(\mathbf{p}) + U(\mathbf{q}) \\
&= \sum_{i=1}^{3} \frac{1}{2m_i}\, p_i^2 - \frac{m_1m_2}{\sqrt{(q_2 - q_1)^2 + \varepsilon^2}} - \frac{m_1m_3}{\sqrt{(q_3 - q_1)^2 + \varepsilon^2/4}} - \frac{m_2m_3}{\sqrt{(q_2 - q_3)^2 + \varepsilon^2/4}}
\end{aligned}$$

is a Hamiltonian function for system (6.8).

Solutions to (6.8) lie in 6-dimensional phase space $\mathbb{R}^3 \times \mathbb{R}^3$. We note, however, that Poincaré simplified his study of the 3-body problem by assuming that one body was small enough so that the gravitational force it exerts can be neglected, and that the two larger masses (called *primaries*) move in circles in a plane $\mathcal{P}$ about their center of mass. This simplified problem is known as the restricted, planar, circular 3-body problem, the goal of which is to understand the behavior of the small body as it moves in $\mathcal{P}$ under the gravitational influence of the primaries.

We follow Poincaré's lead in simplifying the trilinear 3-body problem. As shown in [15], one can assume the center of mass is fixed at the origin (accomplished via a change of coordinates). This implies

$$\frac{m_1q_1 + m_2q_2 + m_3q_3}{m_1 + m_2 + m_3} = 0, \ \text{ that is, } \ m_1q_1 + m_2q_2 + m_3q_3 = 0. \tag{6.9}$$

We also consider a restricted problem: body 3 is so small that the primaries do not feel its gravitational effect. Setting $m_3 = 0$ in (6.9) yields $m_1q_1 + m_2q_2 = 0$. If we assume further that $m_1 = m_2 = m$, then $q_1 = -q_2$ and $p_1 = -p_2$

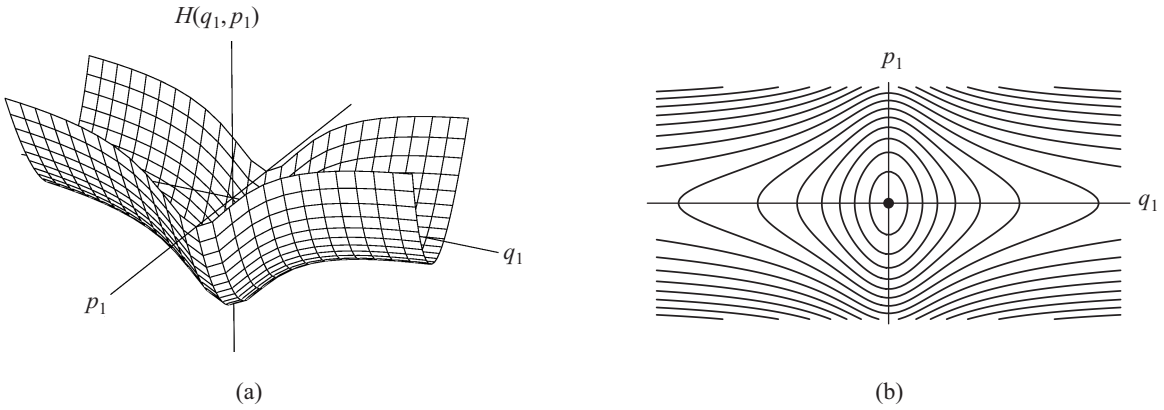


Figure 6.5. (a) The Hamiltonian $H(q_1, p_1) = \frac{1}{2}p_1^2 - 1/(2\sqrt{4q_1^2 + \varepsilon^2})$. (b) The (q_1, p_1)-phase plane (the periodic orbits are moving clockwise). Level sets of the Hamiltonian provide solution curves in the (q_1, p_1)-phase plane.

for all time, so that the positions and velocities of the primaries are always symmetric about the origin. Once we know $q_1(t)$ and $p_1(t)$, we then have $q_2(t)$ and $p_2(t)$. Hence we can reduce (6.8) to the 4-dimensional system (we set $m = 1$ in all that follows)

$$\begin{aligned} &\dot{q}_1 = p_1, \quad \dot{q}_3 = p_3, \quad \text{and} \\ &\dot{p}_1 = -\frac{2q_1}{(4q_1^2 + \varepsilon^2)^{3/2}} \\ &\dot{p}_3 = \frac{q_1 - q_3}{((q_1 - q_3)^2 + \varepsilon^2/4)^{3/2}} - \frac{q_1 + q_3}{((q_1 + q_3)^2 + \varepsilon^2/4)^{3/2}}. \end{aligned} \tag{6.10}$$

Note further that the $\dot{q}_1$- and $\dot{p}_1$-equations in system (6.10) decouple. In addition,

$$H_1(q_1, p_1) = \frac{1}{2}p_1^2 - \frac{1}{2\sqrt{4q_1^2 + \varepsilon^2}}$$

is a Hamiltonian function for this subsystem, leading to the (q_1, p_1)-phase plane sketched in Figure 6.5.

We see that if $p_1(0)$ is small enough, then the motion of m_1 (and so also of m_2) is periodic. We have arrived at the analog of Poincaré's reduction in his study of the 3-body problem: determine the behavior of the small center body as the primaries move periodically about their center of mass.

The periodicity of the primaries lends itself nicely to consideration of a *first return* (or *Poincaré*) map. We will sample the position and velocity of the small body each time the velocity of body 1 equals zero. This yields a mapping P_a, defined on the (q_3, p_3)-plane, and reduces our 6-dimensional system (8) to the study of the orbits of a planar map. (The parameter a represents the amplitude of the large mass oscillation, as discussed in section 6.7.)

Our study will be aided greatly by the use of a symplectic integration algorithm. This type of algorithm is appropriate as the Poincaré map P_a introduced above for the trilinear 3-body problem makes use of a symplectic map in its very definition.

6.4 What is a Symplectic Map?

The approach to symplectic maps taken here, which is ideally suited for undergraduates, follows that found in [21, Ch. 2] and [11, Ch. 6].

Let $\mathbf{v} = (v_1, v_2)^T$ and $\mathbf{w} = (w_1, w_2)^T$ be vectors in the plane. Recall from linear algebra that the area of the parallelogram spanned by $\mathbf{v}$ and $\mathbf{w}$ is $\|\mathbf{v} \times \mathbf{w}\| = |v_1 w_2 - w_1 v_2|$. Symplectic maps are concerned with *oriented* area: the oriented area of the parallelogram spanned by $\mathbf{v}$ and $\mathbf{w}$ is $v_1 w_2 - w_1 v_2$, while the oriented area of the parallelogram spanned by $\mathbf{w}$ and $\mathbf{v}$ is $w_1 v_2 - v_1 w_2$.

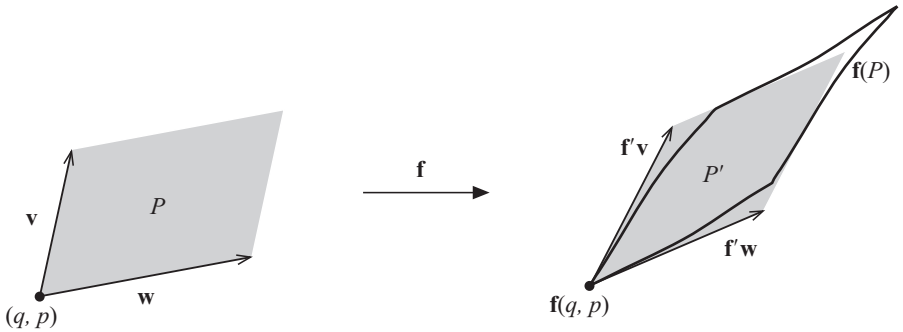


Figure 6.6.

For reasons which will soon become clear, we observe that the oriented area of the parallelogram spanned by $\mathbf{v}$ and $\mathbf{w}$ is given by $\mathbf{v}^T J\mathbf{w}$, where $J = \left[\begin{smallmatrix} 0 & 1 \\ -1 & 0 \end{smallmatrix}\right]$:

$$\mathbf{v}^T J\mathbf{w} = [v_1 \;\; v_2]\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = [v_1 \;\; v_2]\begin{bmatrix} w_2 \\ -w_1 \end{bmatrix} = v_1 w_2 - w_1 v_2.$$

Let $\mathbf{L} : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \times \mathbb{R}$ be a linear map, given by a 2×2 matrix L, and let P denote the parallelogram spanned by two vectors $\mathbf{v}$ and $\mathbf{w}$. Recall that $\mathbf{L}$ maps P to a parallelogram P' spanned by $L\mathbf{v}$ and $L\mathbf{w}$. Then $\mathbf{L}$ preserves oriented area if and only if $(L\mathbf{v})^T J(L\mathbf{w}) = \mathbf{v}^T J\mathbf{w}$ for all $\mathbf{v}$ and $\mathbf{w}$, that is, $L^T JL = J$.

Now let $\mathbf{f} : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \times \mathbb{R}$, $\mathbf{f}(q, p) = (f_1(q, p), f_2(q, p))$, be a differentiable map. Let (q, p) be a point in $\mathbb{R} \times \mathbb{R}$, and let $\mathbf{v}$ and $\mathbf{w}$ be vectors sharing initial point (q, p). The map $\mathbf{f}$ takes the parallelogram spanned by $\mathbf{v}$ and $\mathbf{w}$ to a curved parallelogram $\mathbf{f}(P)$ as in Figure 6.6. We may approximate $\mathbf{f}(P)$ by the parallelogram P' spanned by the vectors $\mathbf{f}'\mathbf{v}$ and $\mathbf{f}'\mathbf{w}$, where

$$\mathbf{f}' = \mathbf{f}'(q, p) = \frac{\partial(f_1, f_2)}{\partial(q, p)} = \begin{bmatrix} \frac{\partial f_1}{\partial q} & \frac{\partial f_1}{\partial p} \\ \frac{\partial f_2}{\partial q} & \frac{\partial f_2}{\partial p} \end{bmatrix}$$

is the Jacobian matrix. This leads to the following definition.

Definition. A differentiable map $\mathbf{f} : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \times \mathbb{R}$ is *symplectic* if

$$\mathbf{f}'(q, p)^T J\, \mathbf{f}'(q, p) = J \;\text{ for all }\; (q, p) \in \mathbb{R} \times \mathbb{R}.$$

By partitioning a given domain in $\mathbb{R}^d \times \mathbb{R}^d = \mathbb{R} \times \mathbb{R}$ with ever-smaller parallelograms, we see that symplectic is equivalent to oriented area preserving when $d = 1$. Though it may seem natural to generalize to "volume preserving" for the case $d > 1$, as we'll see symplectic means more than this.

We introduce the definition for higher dimensions by first considering a linear map $\mathbf{L} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d \times \mathbb{R}^d$ given by a $2d \times 2d$ matrix L. Let I denote the d-dimensional identity matrix, and set $J = \left[\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}\right]$. Given vectors $\mathbf{v}$ and $\mathbf{w}$ in $\mathbb{R}^d \times \mathbb{R}^d$, consider the parallelogram $P = \{ t\mathbf{v} + s\mathbf{w} \;:\; t, s \in [0, 1] \}$. The quantity $\mathbf{v}^T L\mathbf{w}$ represents the sum of the two-dimensional oriented areas of the d parallelograms resulting from the projections of P onto the (q_i, p_i)-coordinate planes, $i = 1, \ldots, d$.

Let's verify this when $d = 2$. Let $\mathbf{v} = (q_1^v, q_2^v, p_1^v, p_2^v)^T$ and let $\mathbf{w} = (q_1^w, q_2^w, p_1^w, p_2^w)^T$. We have

$$\mathbf{v}^T J\mathbf{w} = \begin{bmatrix} q_1^v & q_2^v & p_1^v & p_2^v \end{bmatrix}\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}\begin{bmatrix} q_1^w \\ q_2^w \\ p_1^w \\ p_2^w \end{bmatrix} = \begin{bmatrix} q_1^v & q_2^v & p_1^v & p_2^v \end{bmatrix}\begin{bmatrix} p_1^w \\ p_2^w \\ -q_1^w \\ -q_2^w \end{bmatrix}$$

$$= q_1^v p_1^w + q_2^v p_2^w - q_1^w p_1^v - q_2^w p_2^v = \begin{vmatrix} q_1^v & p_1^v \\ q_1^w & p_1^w \end{vmatrix} + \begin{vmatrix} q_2^v & p_2^v \\ q_2^w & p_2^w \end{vmatrix},$$

as desired.

A differentiable map $\mathbf{f} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d \times \mathbb{R}^d$ is then defined to be *symplectic* if $\mathbf{f}'(\mathbf{q}, \mathbf{p})^T J\, \mathbf{f}'(\mathbf{q}, \mathbf{p}) = J$ for all $(\mathbf{q}, \mathbf{p}) \in \mathbb{R}^d \times \mathbb{R}^d$.

The notion of symplectic for $d > 1$ can be interpreted as follows. Consider an oriented surface Λ in $\mathbb{R}^d \times \mathbb{R}^d$. Let Λ_i denote the projections of Λ onto the (q_i, p_i)-coordinate planes, $i = 1, \ldots, d$. Let Λ_i' denote the projections of $\mathbf{f}(\Lambda)$ onto the (q_i, p_i)-coordinate planes. Using tangent parallelogram approximations, a symplectic map then has the property that the sum of the oriented areas of the Λ_i equals the sum of the oriented areas of the Λ_i'.

We consider an example furnished by the simple harmonic oscillator (6.3). Students of differential equations know the solution to (6.3) satisfying $(q(0), p(0)) = (q_0, p_0)$ is

$$\begin{bmatrix} q(t) \\ p(t) \end{bmatrix} = \begin{bmatrix} \cos \omega t & \frac{1}{m\omega} \sin \omega t \\ -m\omega \sin \omega t & \cos \omega t \end{bmatrix} \begin{bmatrix} q_0 \\ p_0 \end{bmatrix} = L_t \begin{bmatrix} q_0 \\ p_0 \end{bmatrix}, \tag{6.11}$$

where $\omega = \sqrt{k/m}$.

For *fixed* t, the above matrix defines a mapping $\mathbf{L}_t : \mathbb{R} \times \mathbb{R} \to \mathbb{R} \times \mathbb{R}$ which sends the point (q_0, p_0) to the value at time t of the solution to equation (6.2) with initial condition (q_0, p_0).

We claim the map $\mathbf{L}_t$ is symplectic. Since the map is linear (recall t is fixed), the Jacobian matrix at any (q_0, p_0) equals L_t. We compute

$$\begin{aligned} L_t^T J L_t &= \begin{bmatrix} \cos \omega t & -m\omega \sin \omega t \\ \frac{1}{m\omega} \sin \omega t & \cos \omega t \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \cos \omega t & \frac{1}{m\omega} \sin \omega t \\ -m\omega \sin \omega t & \cos \omega t \end{bmatrix} \\ &= \begin{bmatrix} \cos \omega t & -m\omega \sin \omega t \\ \frac{1}{m\omega} \sin \omega t & \cos \omega t \end{bmatrix} \begin{bmatrix} -m\omega \sin \omega t & \cos \omega t \\ -\cos \omega t & -\frac{1}{m\omega} \sin \omega t \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}. \end{aligned}$$

The reader may have noticed that for any t, $\det(L_t) = 1$. This implies the linear map $\mathbf{L}_t$ preserves oriented area which, as mentioned above, is the equivalent of symplectic in $\mathbb{R} \times \mathbb{R}$. For $d > 1$ volume preserving and symplectic are not equivalent. For example, consider the linear map $\mathbf{L}$ defined on $\mathbb{R}^2 \times \mathbb{R}^2$, given by the matrix

$$L = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

The map $\mathbf{L}$ preserves volume in $\mathbb{R}^4$ since $\det L = 1$. It is not symplectic, however, as a simple calculation shows

$$L^T J L = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ -1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} \neq J.$$

The harmonic oscillator example was also chosen to illustrate the fact that "time-t" maps arising from Hamiltonian systems of ODEs are *always* symplectic. We turn now to symplectic maps as the defining characteristic of Hamiltonian systems.

6.5 Hamiltonian Flows and Symplectic Maps

Consider the first-order system of ODEs

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{6.12}$$

where $\mathbf{x} : \mathbb{R} \to \mathbb{R}^n$, $\mathbf{x}(t) = (x_1(t), \ldots, x_n(t))$, and $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ is a C^1 vector field. The *flow* associated with system (6.12) is a function which maps $\mathbf{x}_0$ and a time t to the value at time t of the solution to (6.12) having initial condition $\mathbf{x}_0 = \mathbf{x}(0)$. (We note t must be in the domain of the solution.)

More formally, the flow $\boldsymbol{\phi}(\mathbf{x}, t)$ is a mapping from $\mathbb{R}^n \times \mathbb{R}$ into $\mathbb{R}^n$ satisfying

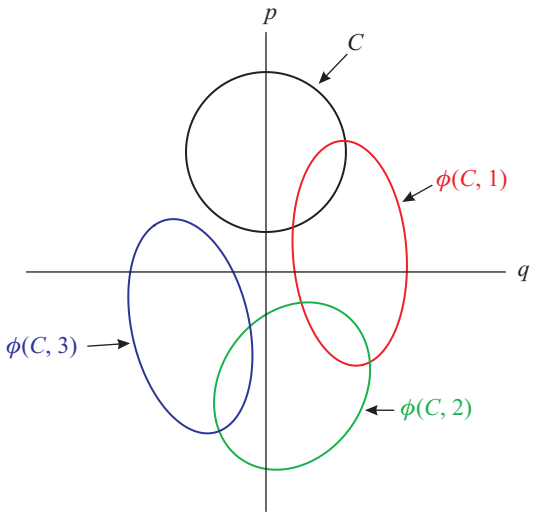


Figure 6.7. The harmonic oscillator flow: the areas enclosed by C and $\boldsymbol{\phi}(C, i)$, $i = 1, 2, 3$, are equal.

(i) $\boldsymbol{\phi}(\mathbf{x}, 0) = \mathbf{x}$ for all $\mathbf{x} \in \mathbb{R}^n$;

(ii) $\boldsymbol{\phi}(\boldsymbol{\phi}(\mathbf{x}, t), s) = \boldsymbol{\phi}(\mathbf{x}, t + s)$ for all $\mathbf{x} \in \mathbb{R}^n$ and all $t, s \in \mathbb{R}$ for which both sides are defined; and

(iii) $\frac{\partial}{\partial t}\boldsymbol{\phi}(\mathbf{x}, t) = \mathbf{f}(\boldsymbol{\phi}(\mathbf{x}, t))$.

The flow is also denoted $\boldsymbol{\phi}_t(\mathbf{x})$. Property (iii) implies that if we hold $\mathbf{x}_0$ fixed, the function $t \mapsto \boldsymbol{\phi}(\mathbf{x}_0, t)$ is a solution of (6.12). Property (i) implies this solution also satisfies the initial condition $\mathbf{x}(0) = \mathbf{x}_0$. Finally, (ii) states that the flow gives rise to a group action (the group $(\mathbb{R}, +)$ is acting on $\mathbb{R}^n$).

We have previously seen an example of a flow, namely that corresponding to the harmonic oscillator and given in equation (6.11). Recall that for each t, $\boldsymbol{\phi}((q, p), t) = L_t(q, p)^T$ preserves area. It is indeed always the case that flows arising from Hamiltonian systems of ODEs preserve area in $\mathbb{R} \times \mathbb{R}$ and, more generally, preserve volume in $\mathbb{R}^d \times \mathbb{R}^d$ phase space. Intuitively, the transport of points in phase space by a Hamiltonian flow is akin to that of particles of an incompressible fluid flow.

More formally, volume preservation means that given any bounded domain $U \subset \mathbb{R}^d \times \mathbb{R}^d$ and any t for which $\boldsymbol{\phi}((\mathbf{q}, \mathbf{p}), t)$ exists for all $(\mathbf{q}, \mathbf{p})$ in U, $\mathrm{vol}(\boldsymbol{\phi}(U, t)) = \mathrm{vol}(U)$ (see Figure 6.7). This fact is a consequence of Liouville's Theorem ([11, p. 228], [12, p. 96]), since the vector field given by system (6.1)

$$\mathbf{f}(\mathbf{q}, \mathbf{p}) = \left(\frac{\partial H}{\partial p_1}, \frac{\partial H}{\partial p_2}, \ldots, \frac{\partial H}{\partial p_d}, -\frac{\partial H}{\partial q_1}, -\frac{\partial H}{\partial q_2}, \ldots, -\frac{\partial H}{\partial q_d}\right)^T$$

has divergence 0 due to the equality of second-order mixed partial derivatives:

$$\operatorname{div} \mathbf{f} = \sum_{i=1}^{d} \frac{\partial^2 H}{\partial q_i \partial p_i} + \sum_{i=1}^{d} \frac{-\partial^2 H}{\partial p_i \partial q_i} = 0.$$

There exist first-order systems of ODEs which are not Hamiltonian, yet for which the associated flow preserves volume. Thus, volume preservation is not a defining characteristic of Hamiltonian systems. Symplecticity, it turns out, is.

Theorem 6.1. [20] *Suppose $H : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a C^2 Hamiltonian function. Then for each fixed t, the corresponding flow $\boldsymbol{\phi}_t$ is a symplectic map wherever it is defined.*

We refer the reader to [11, §6.2] for a proof of Theorem 6.1.

We see that in addition to preserving volume, a Hamiltonian flow must also define a symplectic map for fixed t. It is the converse of Theorem 6.1 which serves to distinguish between these two fundamental properties.

Theorem 6.2. *Suppose $\mathbf{f} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d \times \mathbb{R}^d$ is a C^2 vector field. If the flow corresponding to the system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ is symplectic wherever it is defined, then this system is Hamiltonian.*

We see that symplecticity is both a stronger property than volume preservation and one which completely characterizes Hamiltonian flows.

Though we do not know of an elementary proof of Theorem 6.2, perhaps the $d = 1$ case will provide insight into its veracity. Suppose for fixed t the flow $\boldsymbol{\phi}_t$ associated with the system

$$\begin{aligned} \dot{q} &= f_1(q, p) \\ \dot{p} &= f_2(q, p) \end{aligned} \tag{6.13}$$

is symplectic. Recall that symplectic and oriented area preserving are equivalent in $\mathbb{R} \times \mathbb{R}$. Once again by Liouville's Theorem, the vector field $\mathbf{f} = (f_1, f_2)$ must have divergence 0. Hence $\frac{\partial f_1}{\partial q} + \frac{\partial f_2}{\partial p} = 0$, or $\frac{\partial f_1}{\partial q} = -\frac{\partial f_2}{\partial p}$, which is precisely the condition that guarantees system (6.13) is Hamiltonian in nature [3, §5.3]. A more sophisticated approach is clearly needed for $d > 1$; see [1, §40] or [11, §6.2].

In light of Theorems 6.1 and 6.2, it is no surprise that symplectic maps have great potential when used to numerically integrate Hamiltonian systems of ODEs. In particular, they are ideally suited to generate global, qualitative structure in phase space for the trilinear 3-body problem (Figures 6.9, 6.10, 6.11, and those following). We present one such *symplectic integration algorithm*, along with simulations, in the following sections.

6.6 A Symplectic Integration Algorithm

Standard integration algorithms are very useful when the goal is to obtain short-time, quantitative information. For systems exhibiting chaotic behavior, or fractal structure in phase space, long-time integrations are necessary. For these latter types of dynamical systems, standard integration algorithms are often unreliable.

We do not provide a comprehensive overview of symplectic integration algorithms (SIAs) in this article. There are many different SIAs, and the choice of which to use for a given integration is highly problem dependent. We focus here on one SIA, describe its development, and explain why it works so well. Readers interested in a more thorough treatment of this topic might consult the books [21] or [11], or the article [5].

Consider once again the harmonic oscillator (6.3), setting $k = m = 1$. Recall the total energy is given by $H(q, p) = \frac{1}{2}(p^2 + q^2)$. Euler's method with step size τ, given for this system by the mapping

$$\begin{bmatrix} q_{n+1} \\ p_{n+1} \end{bmatrix} = \begin{bmatrix} q_n + \tau p_n \\ p_n - \tau q_n \end{bmatrix} = \begin{bmatrix} 1 & \tau \\ -\tau & 1 \end{bmatrix} \begin{bmatrix} q_n \\ p_n \end{bmatrix} = L_\tau (q_n, p_n)^T, \tag{6.14}$$

is an integration algorithm which is *not* symplectic. To see this, simply compute

$$L_\tau^T J L_\tau = \begin{bmatrix} 0 & 1+\tau^2 \\ -(1+\tau^2) & 0 \end{bmatrix} \neq J.$$

Alternatively, observe that this algorithm *adds energy* to the system by a factor of $1 + \tau^2$ each iteration (see Figure 6.8):

$$H(q_{n+1}, p_{n+1}) = \frac{1}{2}\left((p_n - \tau q_n)^2 + (q_n + \tau p_n)^2\right) = (1 + \tau^2) H(q_n, p_n).$$

It is this artificial increase (or, for other algorithms, damping) of energy which often makes long-time standard integrations of Hamiltonian systems spurious.

A slight adjustment to mapping (6.14), however, does yield the symplectic map

$$\begin{bmatrix} q_{n+1} \\ p_{n+1} \end{bmatrix} = \begin{bmatrix} 1-\tau^2 & \tau \\ -\tau & 1 \end{bmatrix} \begin{bmatrix} q_n \\ p_n \end{bmatrix} = M_\tau (q_n, p_n)^T. \tag{6.15}$$

A simple computation shows that $M_\tau^T J M_\tau = J$.

Mapping (6.15), of course, no longer preserves the Hamiltonian $H(q, p) = \frac{1}{2}(p^2 + q^2)$. For the Hamiltonian

$$H(q, p) = \frac{1}{2}(p^2 + q^2) - \frac{\tau}{2} pq, \tag{6.16}$$

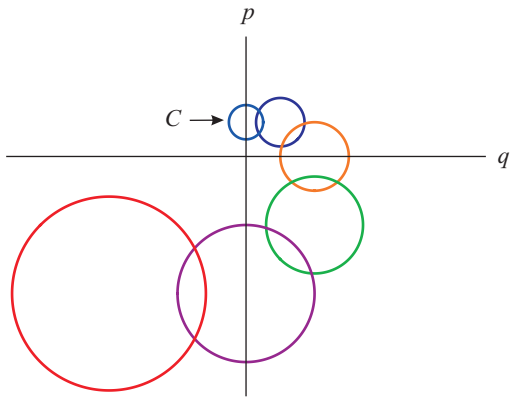


Figure 6.8. Euler's Method is not a symplectic algorithm. Pictured are successive images, moving clockwise, of a small circle C under iteration of Euler's Method for the harmonic oscillator with step size 1.

however, it *is* the case that $H(q_{n+1}, p_{n+1}) = H(q_n, p_n)$, as the reader is invited to check. Hence, map (6.15) *exactly solves* the Hamiltonian system given by (6.16). Starting with initial condition $(1, 0)$, for example, map (6.15) will generate points on the ellipse $\frac{1}{2}(p^2 + q^2) - \frac{\tau}{2}pq = \frac{1}{2}$, which differs from the exact solution $\frac{1}{2}(p^2 + q^2) = \frac{1}{2}$ to system (6.3) only by a term of order 1 in τ. The energy error is bounded for all time (indeed, it is periodic) when using (6.15) to solve system (6.3). This is in contrast to Runge-Kutta type algorithms, for which energy error grows linearly with integration time [8]. For SIAs, energy error is always bounded and periodic, although phase error for periodic orbits can grow linearly with the number of periods [22].

This then is the advantage of an SIA: one uses a symplectic map to exactly solve a Hamiltonian which is close to the original Hamiltonian. While these algorithms have been particularly useful in carrying out long-time integrations of planetary orbits ([18], [23]), they are also used in many other settings ([7], [9], for example).

We now describe the SIA used to generate the images for the trilinear 3-body model in this article. This particular algorithm concerns *separable* Hamiltonians, of which those defining systems (6.3), (6.6) and (6.8) are examples. Separable Hamiltonians are generally of the form $H(\mathbf{q}, \mathbf{p}) = A(\mathbf{p}) + U(\mathbf{q})$, and typically arise in physical systems in which there is no friction.

As this algorithm involves the composition of symplectic maps, we begin by noting that if $\mathbf{f}$ and $\mathbf{g}$ are symplectic maps, then so is $\mathbf{f} \circ \mathbf{g}$:

$$((\mathbf{f} \circ \mathbf{g})')^T \, J \, (\mathbf{f} \circ \mathbf{g})' = (\mathbf{f}'\mathbf{g}')^T \, J \, (\mathbf{f}'\mathbf{g}') = (\mathbf{g}')^T \, (\mathbf{f}')^T \, J \, \mathbf{f}'\mathbf{g}' = (\mathbf{g}')^T \, J \, \mathbf{g}' = J.$$

The basic step of the algorithm consists of the composition of two elementary symplectic maps. We present these maps in the case $d = 1$, that is, for $H(q, p) = A(p) + U(q) : \mathbb{R}^2 \to \mathbb{R}$, in which case our system is

$$\begin{aligned} \dot{q} &= \frac{\partial H}{\partial p} = A'(p) \\ \dot{p} &= -\frac{\partial H}{\partial q} = -U'(q). \end{aligned} \tag{6.17}$$

The basic step easily generalizes to higher dimensional separable Hamiltonian systems.

Fix a step size τ. Let $\boldsymbol{\gamma}_\tau^U(q, p) = (q, p - \tau U'(q))$. Note that $\boldsymbol{\gamma}_\tau^U$ is symplectic as its Jacobian satisfies

$$\begin{bmatrix} 1 & -\tau U''(q) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\tau U''(q) & 1 \end{bmatrix} = \begin{bmatrix} 1 & -\tau U''(q) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -\tau U''(q) & 1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

Similarly, the map $\boldsymbol{\gamma}_\tau^A(q, p) = (q + \tau A'(p), p)$ is symplectic. The composition $\boldsymbol{\gamma}_\tau^A \circ \boldsymbol{\gamma}_\tau^U$

$$\begin{bmatrix} q_n \\ p_n \end{bmatrix} \overset{\boldsymbol{\gamma}_\tau^U}{\rightsquigarrow} \begin{bmatrix} \overline{q}_n \\ \overline{p}_n \end{bmatrix} = \begin{bmatrix} q_n \\ p_n - \tau \left.\frac{dU}{dq}\right|_{q_n} \end{bmatrix} \overset{\boldsymbol{\gamma}_\tau^A}{\rightsquigarrow} \begin{bmatrix} q_n + \tau \left.\frac{dA}{dp}\right|_{\overline{p}_n} \\ p_n - \tau \left.\frac{dU}{dq}\right|_{q_n} \end{bmatrix} = \begin{bmatrix} q_{n+1} \\ p_{n+1} \end{bmatrix}$$

is a symplectic map which serves as the basic step (that is, one step) of the algorithm. This integration method is known as a "drift-kick" method.

For the harmonic oscillator with $k = m = 1$, $A(p) = \frac{1}{2}p^2$ and $U(q) = \frac{1}{2}q^2$. The basic step yields

$$\begin{bmatrix} q_n \\ p_n \end{bmatrix} \overset{\boldsymbol{\gamma}_\tau^U}{\rightsquigarrow} \begin{bmatrix} q_n \\ p_n - \tau q_n \end{bmatrix} \overset{\boldsymbol{\gamma}_\tau^A}{\rightsquigarrow} \begin{bmatrix} q_n + \tau(p_n - \tau q_n) \\ p_n - \tau q_n \end{bmatrix} = \begin{bmatrix} 1-\tau^2 & \tau \\ -\tau & 1 \end{bmatrix} \begin{bmatrix} q_n \\ p_n \end{bmatrix}, \tag{6.18}$$

precisely the map given in (6.15). We see this symplectic basic step provides for an order 1 (in τ) algorithm.

More generally, an integration method is *order r* if the difference between the actual solution values and the approximating values at time t_n are $\mathcal{O}(\tau^r)$ as $\tau \to 0$, for t_n ranging in a bounded interval [21, §3.1].

Symplectic integration algorithms exist for any given even order in τ [25]. The algorithm used for our simulations has order 4, which can be arrived at by manipulating the above basic step as follows.

Definition. A one-step integration algorithm $\boldsymbol{\Psi}_\tau$ is *symmetric* if $\boldsymbol{\Psi}_\tau \circ \boldsymbol{\Psi}_{-\tau} = \mathbf{id}$, where $\mathbf{id}$ denotes the identity map.

It is easy to check that neither Euler's method (6.14) nor the symplectic basic step (6.18), when applied to the harmonic oscillator, is a symmetric method.

One useful property of symmetric integration algorithms is that they must be of even order in the step size [11, §2.3]. We turn the symplectic basic step (6.18) into a symmetric method by setting

$$\boldsymbol{\Gamma}_\tau = \boldsymbol{\gamma}_{\tau/2}^U \circ \boldsymbol{\gamma}_\tau^A \circ \boldsymbol{\gamma}_{\tau/2}^U. \tag{6.19}$$

This algorithm is clearly symmetric as

$$\boldsymbol{\Gamma}_\tau \circ \boldsymbol{\Gamma}_{-\tau} = (\boldsymbol{\gamma}_{\tau/2}^U \circ \boldsymbol{\gamma}_\tau^A \circ \boldsymbol{\gamma}_{\tau/2}^U) \circ (\boldsymbol{\gamma}_{-\tau/2}^U \circ \boldsymbol{\gamma}_{-\tau}^A \circ \boldsymbol{\gamma}_{-\tau/2}^U) = \mathbf{id}.$$

The simple trick (6.19) converts the symplectic basic step (6.18) into an order-2 SIA, albeit one now consisting of three compositions.

Though details fall beyond the scope of this article, Yoshida proved that any symmetric, order-$2n$ symplectic integrator $\boldsymbol{\Psi}_{2n}(\tau)$ can be used to create a symmetric, order $2n+2$ symplectic integrator $\boldsymbol{\Psi}_{2n+2}(\tau)$ [24]. This is accomplished by setting

$$\boldsymbol{\Psi}_{2n+2}(\tau) = \boldsymbol{\Psi}_{2n}(\alpha_1\tau) \circ \boldsymbol{\Psi}_{2n}(\alpha_0\tau) \circ \Psi_{2n}(\alpha_1\tau),$$

where

$$\alpha_0 = \frac{-2^{1/(2n+1)}}{2 - 2^{1/(2n+1)}} \qquad \text{and} \qquad \alpha_1 = \frac{1}{2 - 2^{1/(2n+1)}}.$$

Thus, to convert the algorithm given by basic step (6.19) into an order-4 method, we set $\boldsymbol{\Psi}_\tau = \boldsymbol{\Gamma}_{\alpha_1\tau} \circ \boldsymbol{\Gamma}_{\alpha_0\tau} \circ \boldsymbol{\Gamma}_{\alpha_1\tau}$, with $\alpha_0 = -2^{1/3}/(2 - 2^{1/3})$ and $\alpha_1 = 1/(2 - 2^{1/3})$. Though each step in the algorithm now entails composing nine symplectic maps, it is still relatively easy to program. This is the algorithm executed in *Matlab* and used by the first author to generate our Poincaré map orbits for the trilinear 3-body problem.

6.7 Fractals in the 3-Body Problem

The KAM Theory, named after A. Kolmogorov, V.I. Arnold, and J. Moser, and concerning the dynamics of area preserving perturbations of area preserving maps, provides another framework for the study of the trilinear 3-body problem. This is the approach taken in [15], to which we refer the reader for technical details supporting the observations below.

To a large extent the KAM Theory provides the mathematical underpinnings needed to substantiate Poincaré's fundamental insight that chaotic behavior and fractals can occur in the 3-body problem. We now present examples of this behavior and intricate geometry, via homoclinic points and elliptic island chains, in the figures below.

Recall the Poincaré map P_a for the trilinear 3-body problem introduced in section 6.3. We select an initial condition of the form $(a, 0)$ for the large mass, and plot the position and velocity (q_3, p_3) of the negligible mass each time the velocity of the large mass equals zero.

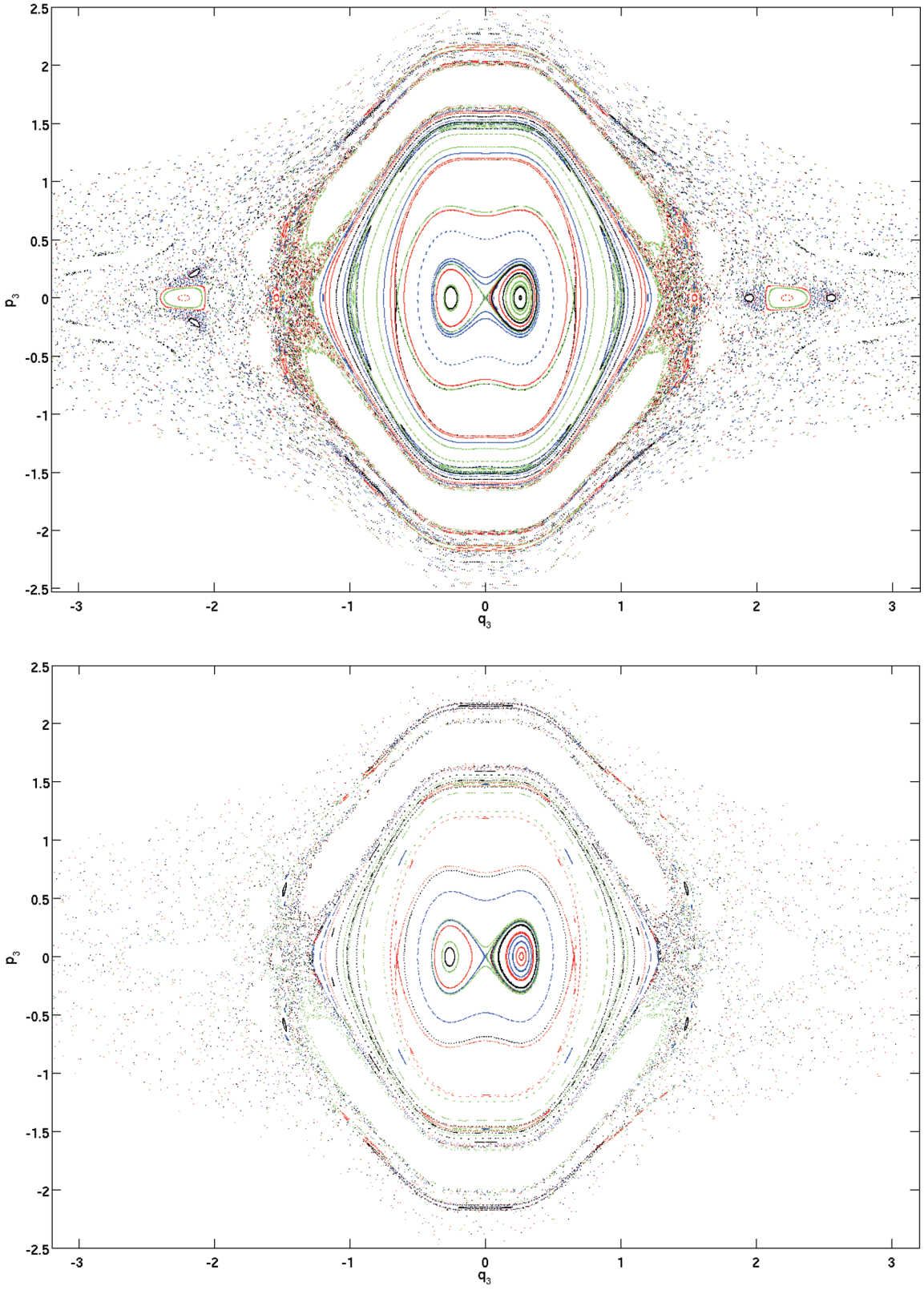


Figure 6.9. Fixed time step ($\Delta t = 0.006$), with the same 83 initial conditions, computing the position and velocity of the small mass out to $t = 2800$ each time the large mass has velocity zero. The parameter $a = 1/(2\sqrt{2})$. Top: fourth order symplectic integration algorithm. Bottom: fourth order Runge-Kutta algorithm.

We note that the (q_1, q_3, p_1, p_3) restricted trilinear 3-body model, in which we set $m_3 = 0$ (equation (6.10)), is no longer Hamiltonian. Recall, however, that the (q_1, p_1) subsystem in equation (6.10) is Hamiltonian. We thus integrate for q_1 and p_1 precisely as described in section 6.6. In this fashion we consider the periodic function $q_1(t)$ to be a known entity. The values for q_1 can then be used to symplectically integrate for q_3 and p_3 in system (6.10). That is, the algorithm first updates the (q_1, p_1)-values, and then integrates for the (q_3, p_3)-values.

Figures 6.9, 6.10 and 6.11 provide striking visual evidence highlighting the superiority of the SIA to the Runge-Kutta algorithm. Both methods used are order 4, and each uses the exact same set of initial conditions, step size, and total integration time. As expected, the SIA provides for a much more detailed rendering of the long-time, global, qualitative structure in the (q_3, p_3)-plane.

What of chaotic behavior? Notice the dense concentration of points roughly near $(1.3, 0)$ and $(-1.3, 0)$ in Figure 6.9. This is evidence of chaotic behavior due to the existence of homoclinic points. Poincaré himself coined the term *homoclinic* for points whose orbits were both forward and backward asymptotic to the same periodic point. He clearly realized that the existence of such points could lead to complicated orbit behavior via "stretching and folding," considered in modern times as the hallmark of systems with chaotic behavior. It was through the work of G. Birkhoff in the early 20th century and S. Smale in the 1960s that chaotic behavior was proven to exist in the presence of homoclinic points, work which involved fractal Cantor sets, symbolic dynamics, and Smale's horseshoe map. These chaotic orbits are prevalent in Figure 6.10.

The KAM Theory can also be used to explain the existence of the fractal structure inherent in nested sequences of *elliptic island chains* [15]. An elliptic periodic point for the map P_a is a periodic point which is surrounded by invariant (under an iterate of P_a), simple closed curves in the (q_3, p_3)-plane. Several elliptic periodic points are evident in Figures 6.9–6.16 via their surrounding invariant curves. (The periodic points are in the centers of these families of curves.)

An elliptic island chain consists of alternating elliptic and saddle periodic points within an annular region. For small area preserving perturbations of area preserving maps, there are infinitely many elliptic periodic points, each surrounded by such island chains, on ever smaller scales ([13], [2, §6.5]).

In Figure 6.12 we see the coexistence of chaotic orbits and a rich, global, qualitative structure. Zooming in near the point $(0.45, 0)$ yields Figure 6.13, where the chains of elliptic islands are clearly evident. A closer inspection of a neighborhood of the origin in Figure 6.12 yields Figure 6.14. The elliptic island chains would continue to appear as one continued to zoom in on various portions of the (q_3, p_3)-plane, providing further evidence of the efficacy of the symplectic integration algorithm in uncovering this fractal structure.

The generation of ergodic orbits, that is, those densely filling regions in the (q_3, p_3)-plane, is also remarkably well done by the SIA. Notice how these orbits clearly avoid the embedded stable regions and elliptic island chains in Figures 6.15 and 6.16.

6.8 Conclusion

From an historical perspective, the n-body problem is the central problem in the field of dynamical systems. Via the trilinear 3-body problem, we have been able to introduce undergraduates to some of the beautiful mathematics and remarkable images generated through its study. We have provided a relatively elementary introduction to the notion of symplectic maps, their relationship to Hamiltonian systems of ODEs, and their use as the basis for a symplectic integration algorithm. We have used the SIA to unearth fractal structure and chaotic dynamics in the trilinear 3-body problem, as Poincaré would have done had he access to our modern tools.

The SIA is clearly superior to the Runge-Kutta algorithm when integrating Hamiltonian systems of ODEs as it is designed to conserve energy (see Figure 6.17). This implies the SIA is ideally suited to generate long-time integrations, which are necessary if one is interested in the global qualitative structure of phase space. New insights into the dynamics of the n-body problem and fractal structure in the phase space will continue to be revealed through the ongoing development and use of symplectic integration algorithms.

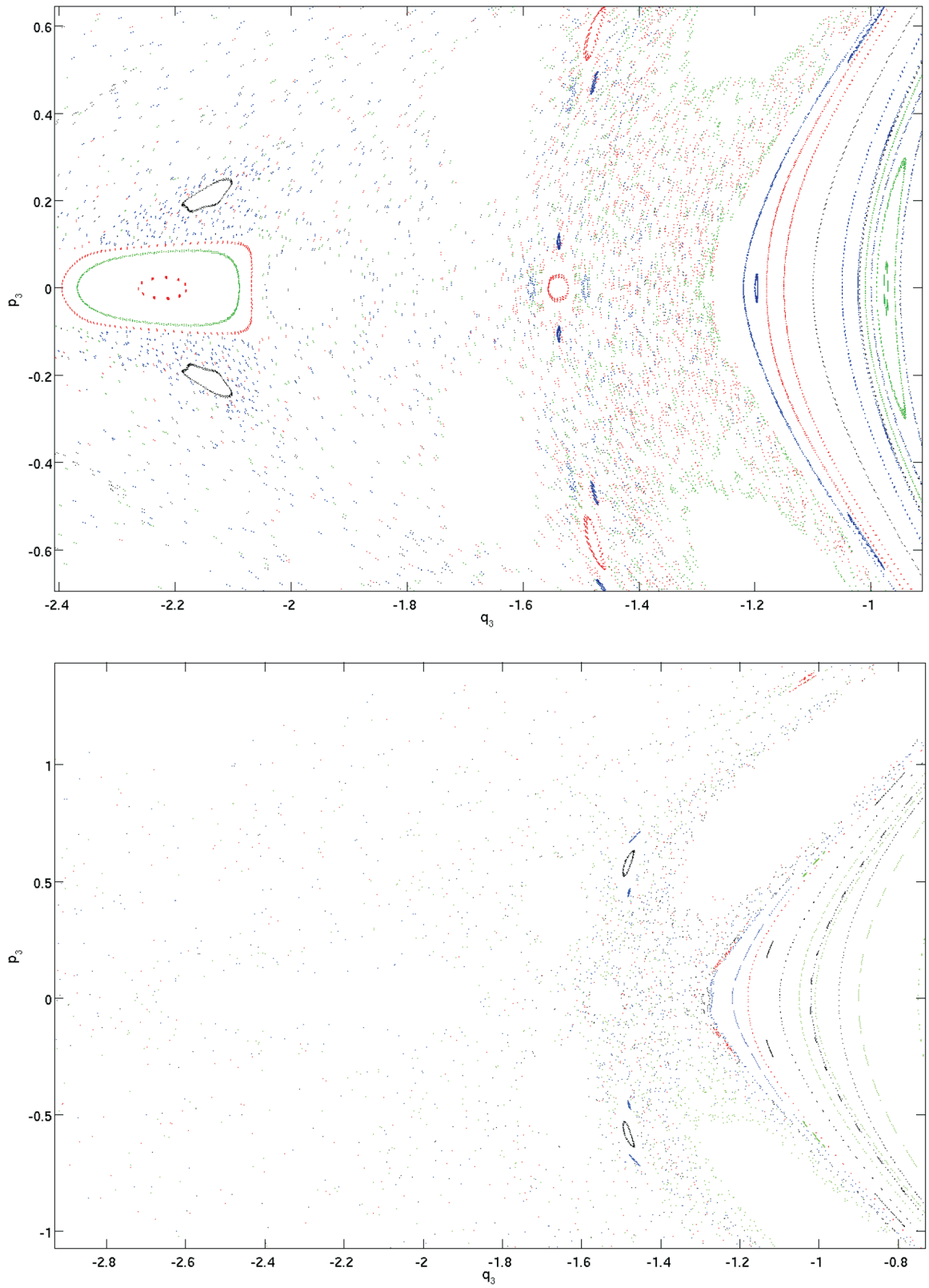


Figure 6.10. As in Figure 6.9. Top: fourth order symplectic integration algorithm. Bottom: fourth order Runge-Kutta algorithm.

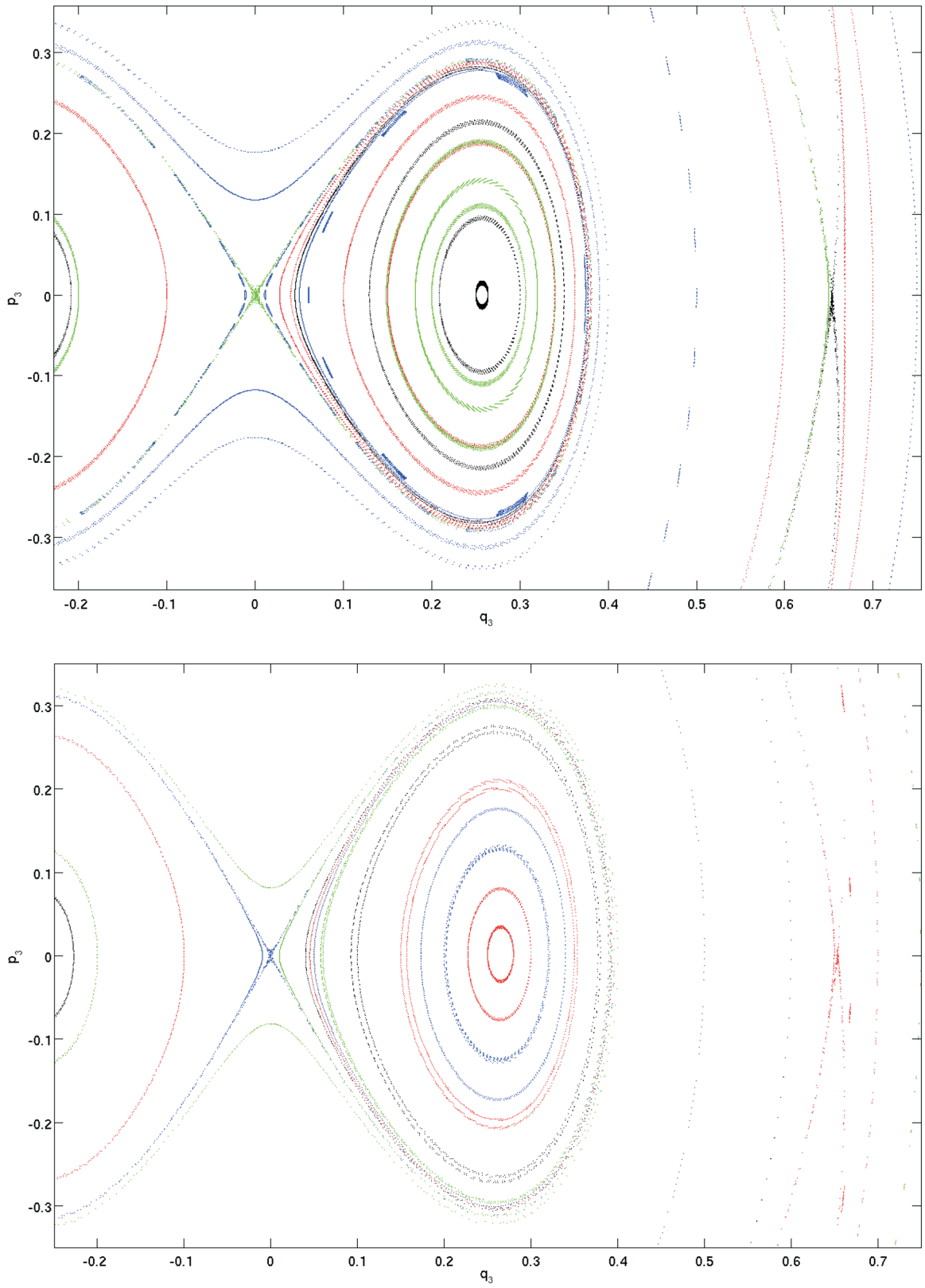


Figure 6.11. As in Figure 6.9. Top: fourth order symplectic integration algorithm. Bottom: fourth order Runge-Kutta algorithm.

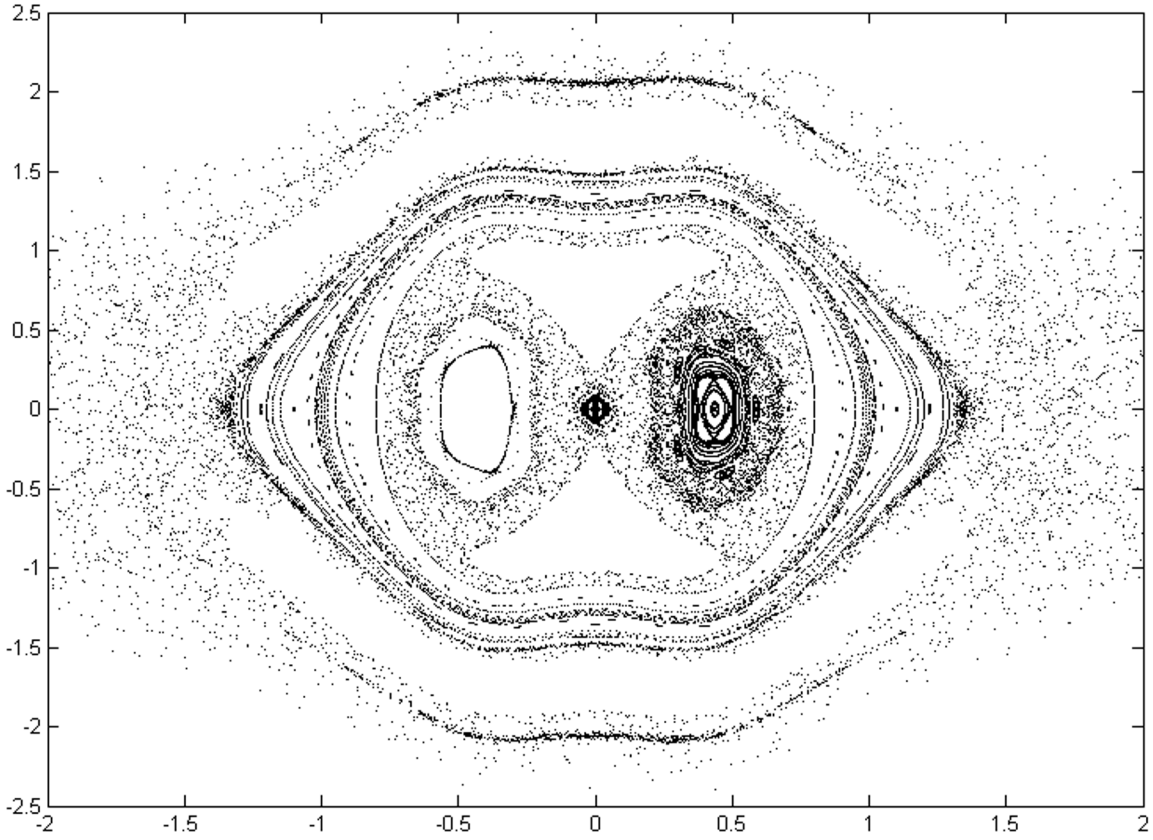


Figure 6.12. Orbits of the Poincaré map P_a, $a = 0.4325$ (300,000 iterates for each of roughly 100 initial conditions).

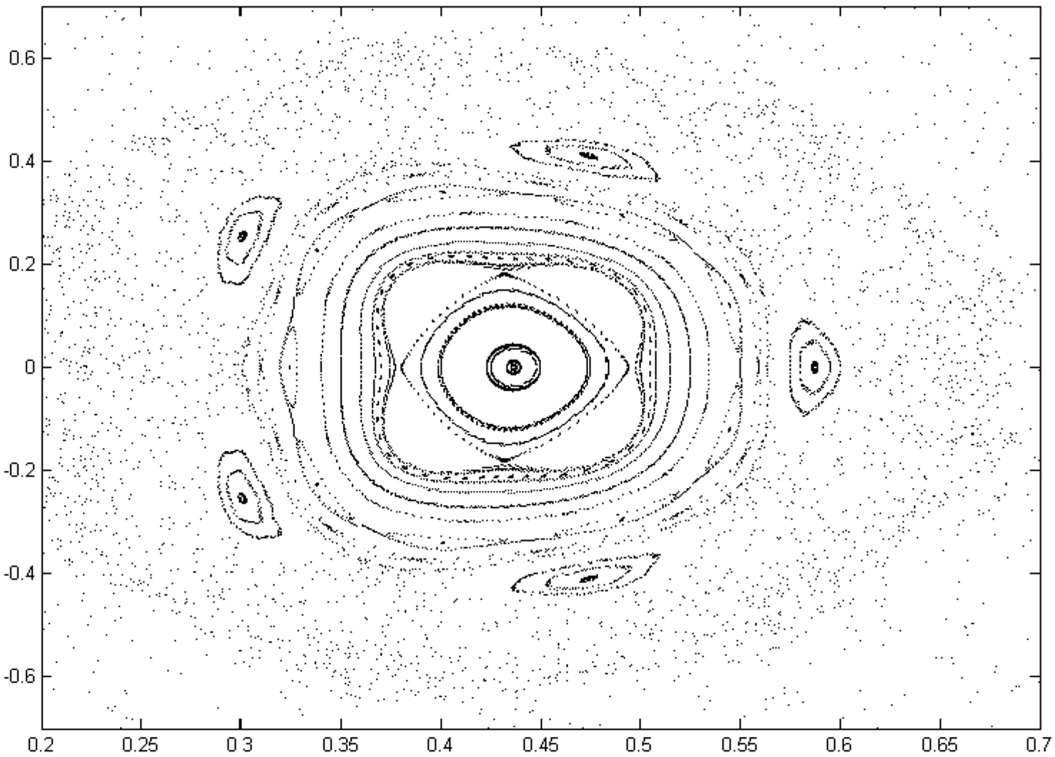


Figure 6.13. Zooming in on the region near $(0.45, 0)$ in Figure 6.12. Note the many elliptic island chains.

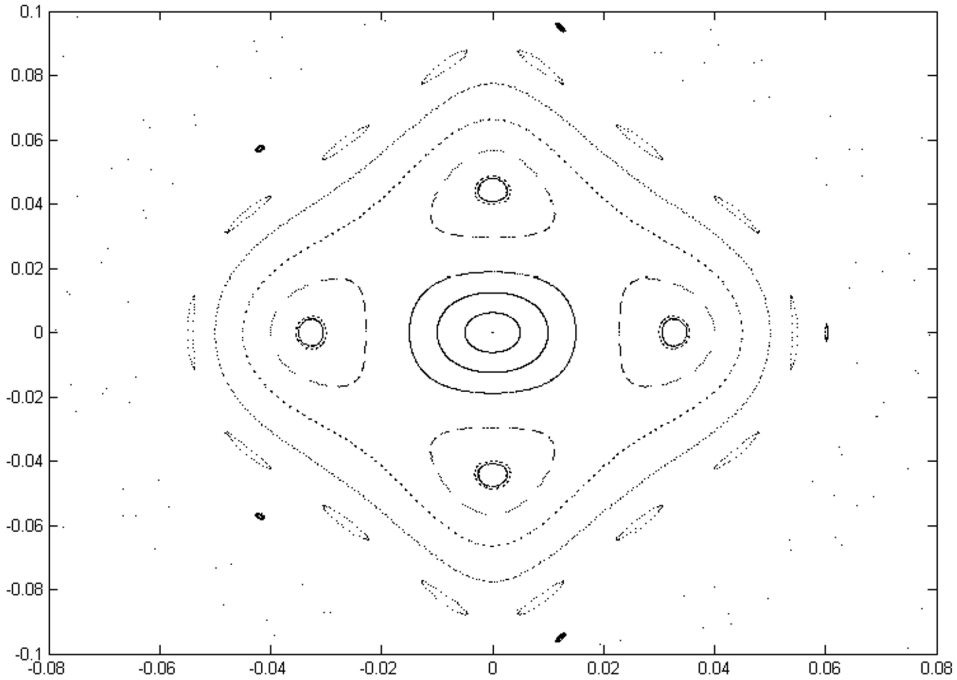


Figure 6.14. P_a-orbits in a neighborhood of the origin in Figure 6.12.

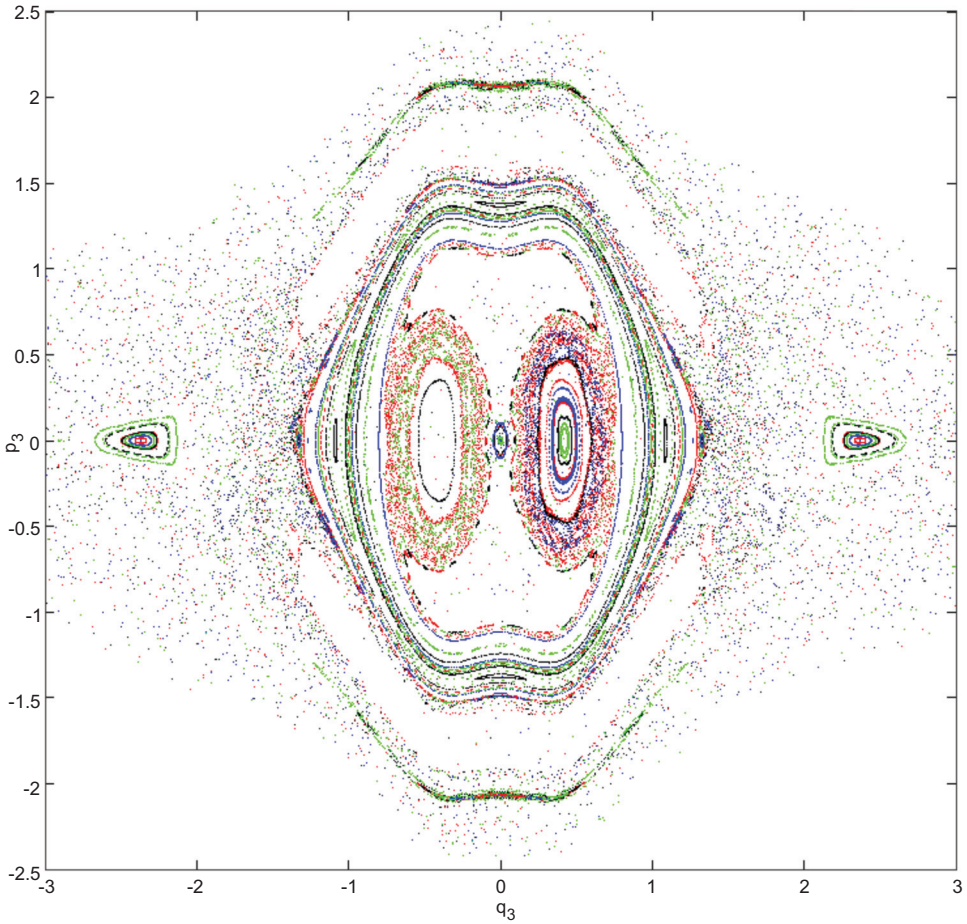


Figure 6.15. The ergodic orbits of P_a avoid the stable regions ($a = 0.425$).

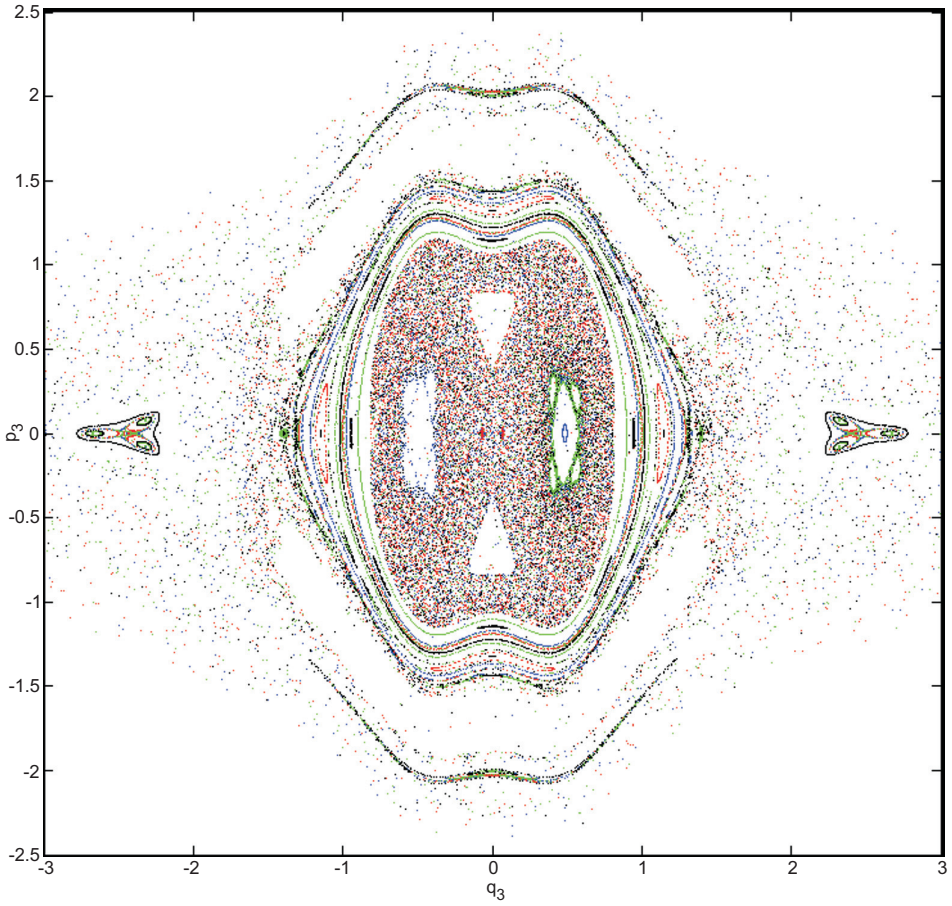
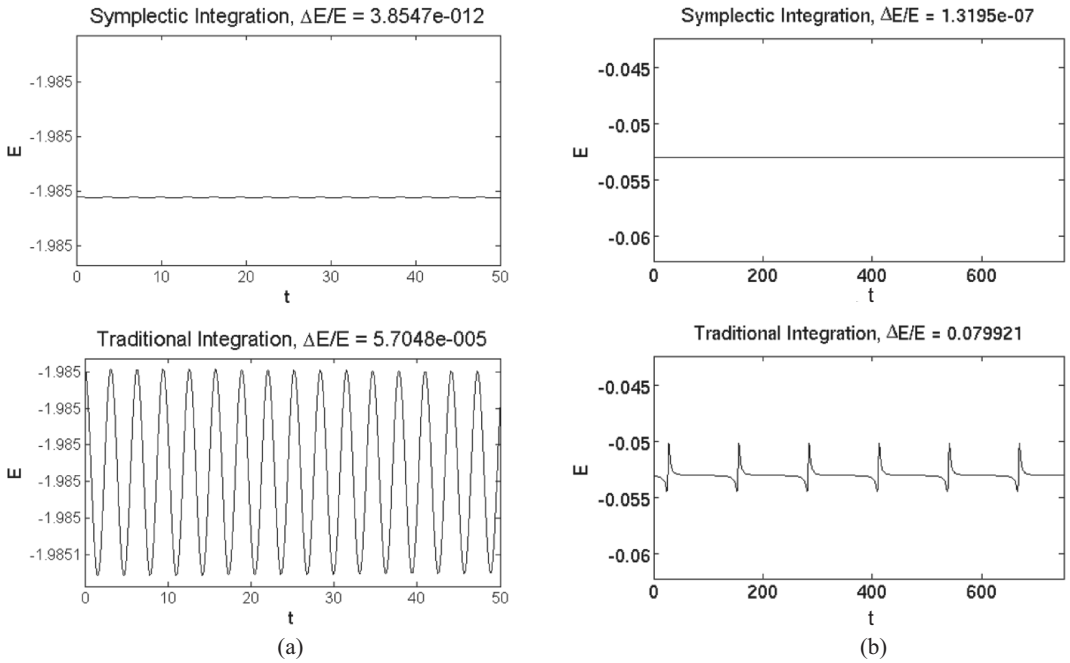


Figure 6.16. The ergodic orbits of P_a are well-rendered by the SIA ($a = 0.46$).



(a) (b)

Figure 6.17. Comparison of energy conservation between symplectic integration (top) and Runge-Kutta (bottom). (a) For the ideal pendulum. (b) For the two-dimensional 2-body problem.

Bibliography

[1] Arnold, V.I., *Mathematical Methods of Classical Mechanics*, New York: Springer-Verlag, 1989.

[2] Arrowsmith, D.K. and C.M. Place, *An Introduction to Dynamical Systems*, Cambridge: Cambridge University Press, 1990.

[3] Blanchard, P., R.L. Devaney, and G.R. Hall, *Differential Equations*, Pacific Grove, CA: Brooks/Cole, 2006.

[4] Borrelli, R. and C. Coleman, *Differential Equations A Modeling Perspective*, New York: Wiley, 2004.

[5] Channell, P.J. and C. Scovel, "Symplectic integration of Hamiltonian systems," *Nonlinearity*, 3(1990), 231–259.

[6] Chin, S.A., "A fundamental theorem on the structure of symplectic integrators," *Physics Letters A*, 354(2006), 373–376.

[7] Chin, S.A. and C. Chen, "Fourth order gradient symplectic integrator methods for solving the time-dependent Schrödinger equation," *J. of Chemical Physics*, 114(17), (2001), 7338–7341.

[8] Chin, S.A. and D. Kidwell, "Higher-order force gradient symplectic algorithms," *Physical Review E*, 62(6), (2000), 8746–8752.

[9] Creutz, M. and A. Gocksch, "Higher-order hybrid Monte Carlo algorithms," *Physical Review Letters*, 63(1), (1989), 9–12.

[10] Forest, E. and R. Ruth, "Fourth-order symplectic integration," *Physica D*, 43(1990), 105–117.

[11] Hairer, E., C. Lubich, and G. Wanner, *Geometric Numerical Integration*, New York: Springer, 2006.

[12] Hartman, P., *Ordinary Differential Equations*, Boston: Birkhäuser, 1982.

[13] Hénon, M., "Numerical exploration of Hamiltonian systems," in *Comportement Chaotique des Systèmes Déterministes: Les Houches, Session XXXVI, 1981*, G. Iooss, R.H.G Helleman, & R. Stora (eds.), New York: North Holland, 1983.

[14] Hirsch, M., S. Smale, and R.L. Devaney, *Differential Equations, Dynamical Systems, & an Introduction to Chaos*, San Diego: Academic Press, 2004.

[15] Lodge, G., J.A. Walsh, and M. Kramer, "A trilinear three-body problem," *International Journal of Bifurcation & Chaos*, 13(8), (2003), 2141–2155.

[16] Meyer, K. and G.R. Hall, *Introduction to Hamiltonian Dynamical Systems and the N-body Problem*, New York: Springer-Verlag, 1992.

[17] Moore, C. and M. Nauenberg, "New periodic orbits for the n-body problem," *Journal of Computational and Nonlinear Dynamics*, 1(4), (2006), 307–311.

[18] Murray, N. and M. Holman, "The origin of chaos in the solar system," *Science*, 283(1999), 1877–1881.

[19] Poincaré, H., "Sur le problème des trois corps et les équations de la dynamique," *Acta*, 13(1890), 1–270.

[20] ——, *Les Méthodes Nouvelles de la Mécanique Céleste. Tome III*, Paris: Gauthiers-Villars, 1899.

[21] Sanz-Serna, J.M. and M.P. Calvo, *Numerical Hamiltonian Problems*, London: Chapman & Hall, 1994.

[22] Scuro, S. and S.A. Chin, "Forward symplectic integrators and the long-time phase error in periodic motions," *Physical Review E*, 71(2005), 056703-1–056703-12.

[23] Sussman, G. and J. Wisdom, "Chaotic evolution of the solar system," *Science*, 257(1992), 56–62.

[24] Yoshida, H., "Construction of higher order symplectic integrators," *Physics Letters A*, 150(1990), 262–268.

[25] ——, "Symplectic integrators for Hamiltonian systems: basic theory," in *Chaos, Resonance, and Collective Dynamical Phenomena in the Solar System: Proceedings of the 152nd Symposium of the International Astronomical Union*, S. Ferraz-Mello (ed.), Boston: Kluwer, 1992.

About the Editors

Denny Gulick received his BA at Oberlin College, MA and PhD at Yale University. After teaching two years at the University of Pennsylvania, he moved to the University of Maryland, where he has taught in the department of mathematics for a long time. He received the College Dean's Award for Excellence in Teaching in 1992 and the Kirwan Undergraduate Education Award in 2000. In addition to membership in the MAA, he is the organizer and chair of the (Maryland) Statewide Mathematics Group. Among his publications are articles in functional analysis, and textbooks *Calculus with Analytic Geometry* (Editions 1–5) and *Calculus*, Edition 6, as well as *College Algebra and Trigonometry*, Editions 1–3, all with co-author Robert Ellis. In addition, he published *Encounters with Chaos* and *Encounters with Fractals*, the latter privately published. Finally, he and Jon Scott have joined together in producing a series of summer workshops on Chaotic Dynamics and Fractal Geometry, as well as several contributed paper sessions on fractals and chaos at the Joint Mathematics Meetings.

Jon Scott received a BS from the University of Albany; and an MA and SpA from Western Michigan University. He is Professor of Mathematics at Montgomery College in Montgomery County, Maryland. An active MAA member with a long-time interest in professional development, he is a member of the team that conceived and manages the MAA Professional Enhancement Program (PREP), and is a former visiting mathematician at the MAA. He received an Outstanding Faculty Award from Montgomery College in 2003 and the Certificate of Meritorious Service from the MAA in 2005. Together with Denny Gulick, he led the *Maryland Undergraduate Mathematics Enhancement Program, A Regional Coalition to Promote Visual Thinking in Mathematics*, a series of summer workshops focusing on Chaotic Dynamics and Fractal Geometry. Gulick and Scott have also organized several contributed paper sessions on fractals and chaos at the Joint Mathematics Meetings.