The Chaos Cookbook

a practical programming guide

The Chaos Cookbook

a practical programming guide

Joe Pritchard

BUTTERWORTH
HEINEMANN

An imprint of Butterworth-Heinemann Ltd
Linacre House, Jordan Hill, Oxford OX2 8DP

PART OF REED INTERNATIONAL BOOKS

OXFORD LONDON BOSTON MUNICH NEW DELHI
SINGAPORE SYDNEY TOKYO TORONTO WELLINGTON

First published 1992

This book was typeset and designed by M. A. McKenzie, is set in Donald E. Knuth's Computer Modern Roman 10pt typeface and was reproduced from camera-ready copy generated by TEX and LATEX.

Printed and bound in Great Britain

ISBN 0-7506 0304-6

Library of Congress Cataloging-in-Publication Data
Available from the publishers

British Library Cataloguing in Publication Data
Available from the publishers

1 2 3 4 5 96 95 94 93 92

Preface

The subject of chaos and fractals has come into popular prominence in the last few years due in the main to the fact that the fractals can create stunning graphical images on computers. Indeed, many magazines have carried programs to generate what is probably the hallmark of the field, the Mandelbrot set. However, beyond the aesthetic pleasure of these images the science of chaos has implications for us all, in areas as diverse as next weeks weather, Jupiter's famous red spot and heart attacks and was actually born thirty years ágo! Chaos is likely to cause as much change in the way that scientists think of the world as did the replacement of Newton's classical mechanics with the quantumn theory. Despite this, the basic principles are breathtakingly simple, and anyone with a home computer can get involved and conceivably make a contribution to this still young science.

I hope that this book will provide food for thought for computer hobbyists, students and anyone interested in the subject, and may provide a useful starting point before going in to the more specialist works in the field. The idea of this book is to provide a 'cookery book' for various chaotic systems, and, like all recipes, experimentation will often give new and unexpected results. So, use these programs as starting points for your own work.

Personally, my interest in this area goes back about seven years to when I came across Koch snowflakes for the first time on a small computer. After a while, I encountered the Mandelbrot set, the Lorenz equations and other aspects of the world of chaos, and began to write programs to generate the images seen in books and magazines. And that's where the problems started; I could find nothing to bridge the gap between the pretty pictures and the mathematics. This book, I hope, will provide a practical guide to bridge that gap, and to introduce the subject of chaos and fractals, using small computers as a laboratory for exploring the beautiful and fascinating world on the boundary of order and chaos.

No book is created without a team effort; in this case I'd like to particularly mention Andrew Parr and Peter Dixon, my publisher, for their encouragement and support on this project.

Finally, special thanks must go to my wife, Nicky, for support for what

has probably appeared to be a rather odd interest of mine, and to all those people who've given me advice and inspiration during the preparation of this book. Thanks folks!

I would like to dedicate this book to the memory of
Julia Helen Crick,
a dearly loved and greatly missed friend.

JOE PRITCHARD
Sheffield
1991

Contents

Introduction

This book is about chaos, a relatively new science, in which the laboratory equipment consists of computer software, the results are often graphical displays of stunning complexity and the subjects of the experiments are numbers themselves. Indeed, many people have already suggested that the term 'Experimental Mathematics' be applied to the whole area of chaos theory and fractals. In this section of the book, I want to look briefly at how this book can be used, and at some of the techniques that we'll be employing.

I suppose that before going any further we should bring in a definition unfortunately, chaos doesn't lend itself to rapid definitions, so I'll be giving a fuller one later in the book. However, for the time being let's just say that a chaotic system is a deterministic system with great sensitivity to initial conditions. In other words, the behaviour of such a system is predictable by the use of suitable mathematical equations *but varies greatly depending upon what the starting conditions are.* Thus a model of the world's weather on computer (Chapter 4, The Lorenz equations) can give staggeringly different results when the value of an input parameter is varied by, say, 1 part in 1000. Intuition tells us that small changes in input should give proportional changes at the output; chaos shows us that this isn't necessarily the case.

This book is aimed at anyone with access to a computer running Turbo Pascal or BBC BASIC, although the programs in the book are presented in such a way that they can be duplicated in virtually any language you care to choose, provided that the language has the means of drawing graphical images on a computer display. The programs in the book are designed as starting points for further experimentation, and I'll be giving hints of further things to try in each chapter. It's hoped that the listings will give you a starting point for your own programs.

Although the subject is based in maths, this book is designed as a 'how to do it' book, so I've kept the mathematical content to a minimum. For those of you wishing to follow up some of the mathematics, I've included a substantial list of further reading and computer software. In addition to the computer-based work, I've included in Chapter 11 guidelines for some

experimental work in chaotic systems that you might find in the real world. As to actually using this book, the newcomer might like to start at Chapter 1 and work on through; those with some knowledge of chaos might prefer to get right on with the programs featured. Finally, if there's enough interest, who knows; there might be a book called *Advanced Chaos* ...

The computer

To get the best out of this book you need a computer of some sort. I have concentrated on the IBM PC and clones and the BBC Model B. In order to save space, from here on I'll refer to the BBC Model B as the BBC Micro, and to the IBM PC and its clones as a PC. The main points about any computer used for chaos experiments are as follows.

Graphics

The computer needs some medium-to-high-resolution graphics capability to display the images generated. If you have a PC, then the only problem you'll encounter is if you've got a machine fitted with a text-only display adapter. Most graphics systems fitted to computers are colour, but this doesn't matter too much; very attractive monochrome images can be created.

More important than colour is the resolution of the graphics display. This is the number of individually addressable points available on the screen. I would say that a minimum requirement for the work featured in this book is about 200 horizontal by 200 vertical points (*pixels*) (200×200). You may find that some authors suggest a higher resolution graphics screen than you have on your machine; if this is the case, then my advice would be to try out the programs on your computer unless the author says the program definitely *won't* work. After all, experimentation is the spice of chaos!

Even if your machine cannot support graphics, some chaos work can be done with a text-only display, by printing out the values rather than graphing them. An example of this is given in Chapter 4 when the Lorenz equations are discussed.

The BBC and IBM PC are two machines capable of supporting good graphics. Here follows a brief description of the graphics modes available on the BBC and the common graphics standards for the PC.

BBC graphics modes

On the BBC Microcomputer, there is a trade-off between screen resolution and memory available for programs. Table 1 shows some common BBC

graphics modes. Modes 0 to 2 take up 20k of memory, and this leaves only around 5k or so for programs when the BBC has a disc drive fitted. Modes 4 and 5 take up 10k of memory.

Mode	Colours	Resolution
0	2 colours	640×256 pixels
1	4 colours	320×256 pixels
2	16 colours	160×256 pixels
4	2 colours	320×256 pixels
5	4 colours	160×256 pixels

Table 1. BBC Micro screen modes.

The other BBC screen modes are not really suitable for use in most of the programs in this book, but can be pressed into service for things like the Life program in Chapter 10.

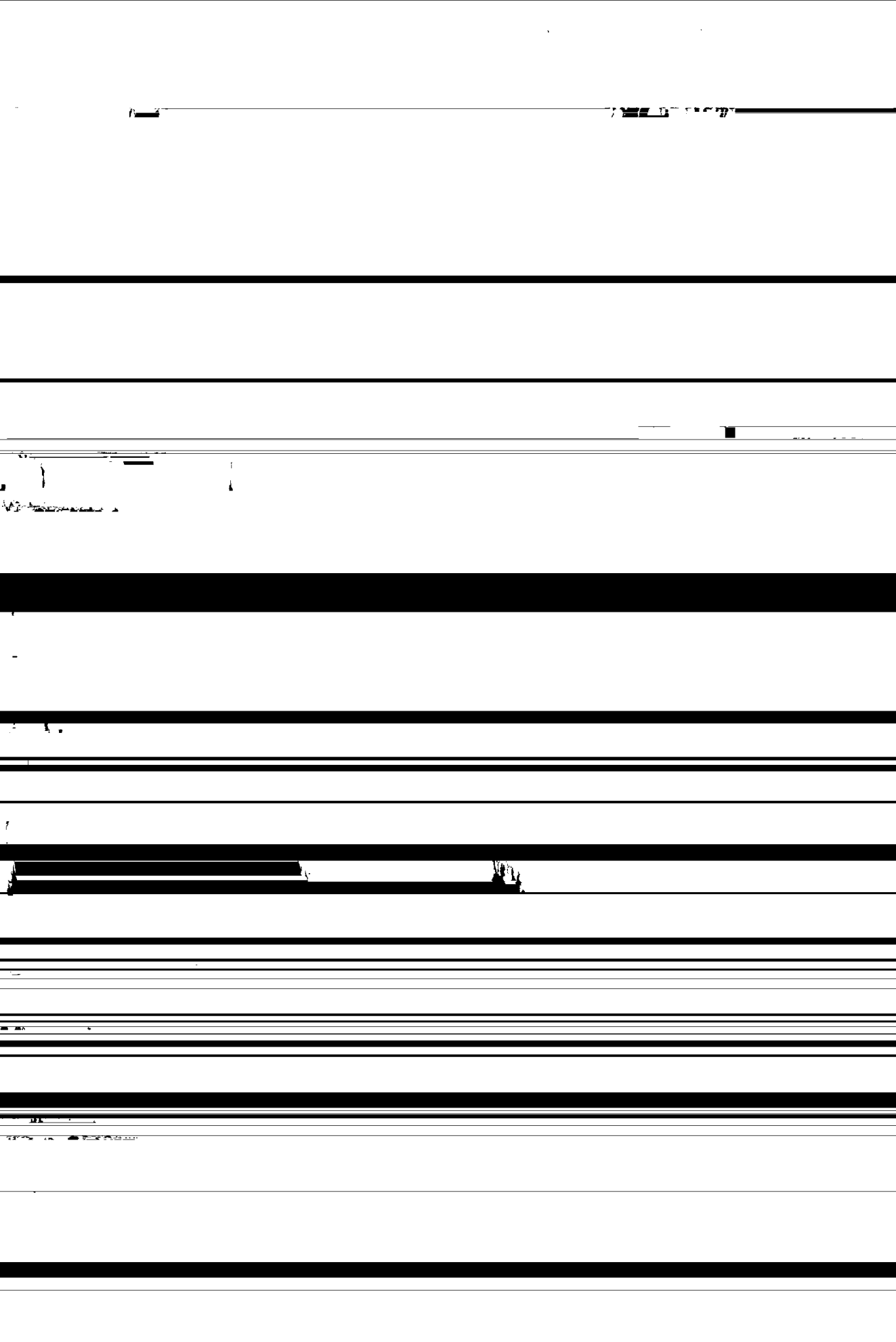
IBM graphics adapters

PCs don't have a range of built-in graphics modes like the BBC; instead, they can be equipped with an add-in card and suitable display to suit the requirements of the user. Once upon a time, all PCs came with *no* display

Adapter	Resolution	Colours	BIOS Mode
CGA	40×25 and 80×25	16 colour text	
	320×200 pixels	4 colours	4 and 5
	640×200 pixels	2 colours	6
EGA	as CGA plus:		
	320×200 pixels	16 colours	13
	640×200 pixels	16 colours	14
	640×350 pixels	16 colours	15
VGA	as EGA plus:		
	640×480 pixels	2 colours	17
	640×480 pixels	16 colours	18
	320×200 pixels	256 colours	19

Table 2. Common IBM PC screen modes.

system at all; a shock to those of us who were more used to home computers! The most common graphics modes supported on IBM machines are shown in Table 2.



actual processing in programs, leaving the BBC's own 6502 to look after just the keyboard and screen.

Life is more complicated with the IBM PC. There are three CPU chip families in wide circulation.

8088/8086: These chips are quite old, but are still to be found in many machines, such as the Amstrad PC1512, Toshiba 1200 and many other low-cost clones. Machines using these chips are often called PC machines, as the original IBM PC used one of these processors.

80286: This is a more powerful processor, which also can handle more computer memory than the 8086/8088 processor. Machines using this processor are often called AT clones.

80386/80386SX: These processors are very powerful indeed, and PC clones using these machines represent the most powerful machines in common use. The SX chip is a slightly cut-down version of the 80386, but for our purposes the SX chip is quite adequate.

As well as the processor type, the speed at which it operates is also quite important. For example, the BBC 6502 trundles along at 1MHz (1 million steps per second). This isn't the same as 1 million calculations every second. The 8086/8088 processors usually operate in the 4MHz to 8MHz range, 80286s up to 12MHz and 386SX/386 processors anywhere between 15 and 33MHz. You can get 20MHz 80286 chips which will outpace some of the slower 80386SX chips, but do not offer some of the more subtle advantages of the 80386SX. However, for chaos and fractals work, these chips are fine. Some machines are equipped with a 'Turbo' mode, in which a machine that normally runs at, say, 4MHz can, at the push of a button, run at 7 or 8MHz. The reason for having two modes is that some software requires that the computer be running at an official speed—in this case, the 4MHz speed—and may not function correctly if the higher speed is used.

A useful adjunct to the CPU for any computer that's doing a lot of numerical work is a special chip called a ***maths coprocessor.*** In the usual run of things, the CPU has to do all the arithmetic, and while it's doing this it can't do anything else. Mathematical operations are not usually the forte of computer chips; they're fine at doing simple arithmetic, but to do complex mathematics they have to execute programs of instructions and this takes time. A coprocessor is designed to do the complicated mathematical operations under the control of the CPU, and it will then pass the results back to the CPU when the calculations are completed. Such a device will make vast improvements in speed to the overall running of mathematically oriented programs, provided that the language used to write the programs can support the maths coprocessor. There is no dedicated maths coprocessor available for the 6502 chip, but for the others, the processor number

needed is obtained by replacing the 6 in the part name with a 7. So the 80286 requires an 80287 maths coprocessor.

Disc drives

The computer disc drive system is simply used for rapid storage and retrieval of data. It's not essential for chaos work; you can use cassette tapes for program and data storage on the BBC Micro, but it makes life easier! Programs can be retrieved more quickly from disc than from tape. On the PCs, there isn't an effective tape system available, and all clones need at least one disc drive, as described below. Furthermore, on PCs many computer languages use the disc as temporary storage whilst you are writing programs, so the disc drive effectively becomes an extension of your computer's memory.

BBC disc systems

There are two main systems on the BBC Micro. These are the 40-track disc, capable of holding 100,000 characters of data (100k) and the 80-track disc, which holds 200k. Figure 1 shows a typical *floppy* disc. The disc fits into a disc *drive* which allows data to be read from and written to the disc as a series of magnetic imprints on the disc surface. The discs are reusable and if you need to use another disc, you simply take the current disc out of the drive, put it in a storage bag, and stick in another one.

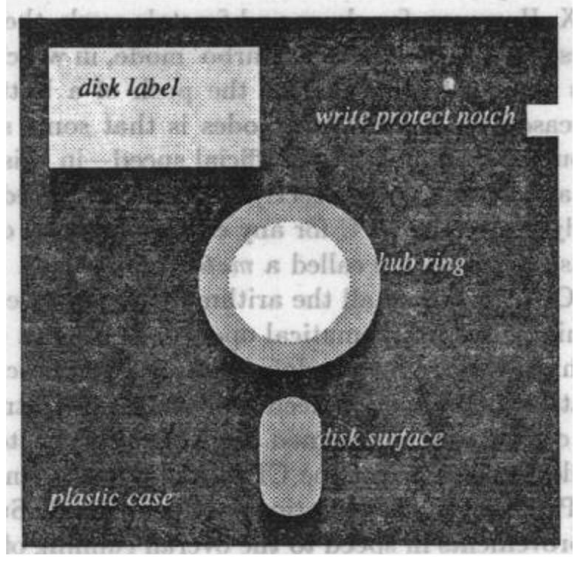


Figure 1. A typical floppy disc.

Floppy discs come in two physical sizes (5.25" and 3.5") and the amount of data that can be stored on them depends upon how the computer *formats* the disc to receive information.

PC disc systems

On PCs there are a variety of floppy disc formats to deal with:

3.5" There are two types of 3.5" disc, called *low density* and *high density*. Low density discs can hold about 720k of data, and high density discs hold about 1400k of data.

5.25" Again, high and low density discs are available, with low density discs holding 360k of data and high density discs holding 1200k of data.

Hard discs

All PC clones will come with at least one floppy disc drive. Some machines have two, but a more common configuration is to have one floppy disc drive and one hard disc. A hard disc is a *non-removable* storage medium; that is, when it fills up you have to erase some old data before you can save new material on to the disc. It's not easy to remove the disc and stick a new one in, but the disc is reusable, and many users move old data from their hard disc on to floppy discs, and then erase the old data from the hard disc to make space for new material. Hard discs can hold large amounts of data—anywhere between 20 million and several hundred million bytes—and are also much faster than floppy discs. A hard disc is well worth the extra cost as it usually makes programming easier and faster and is particularly valuable for storing and recovering screen image files or other large amounts of data.

Interfaces

Computer interfaces are the ways in which the computer talks to other devices in the outside world. Most computers come with at least an interface to allow the computer to send information to a printer to produce hard-copy results from programs. This is usually what is known as a *Centronics* interface, which is a standard means of communication between computers and printers. A further means of communication between computers and peripheral devices is via an *RS232* interface, which allows the computer to communicate with plotters, which can draw graphs in many colours, or allows data transfer between two different computers, even if they are of different makes. If you have an additional piece of equipment called a *modem*, you can communicate with other computers over the telephone line via the RS232 interface.

If you go on to experiment with chaotic systems in the real world, rather than various computer models, then you will require additional interfaces on your computer to convert things happening in the real world into electrical signals that the computer can understand. The BBC Micro is equipped with some of these interfaces, in the form of a *User Port* and *Analogue to Digital Converter* (ADC). The User Port allows simple electrical on or off signals to be read by the computer, whilst the ADC allows continuously varying voltages to be recorded on a computer. Quantities such as light level or temperature can thus be recorded on a computer via a suitable electronic circuit to convert, say, light levels into voltages.

Programming languages

Without suitable programs, computers are simply hi-tech door stops. These programs—sequences of instructions for the computer to follow to get a particular result—are written in particular computer languages. The CPU of a computer only understands one language. This is called machine code, and machine code programs are incredibly fast because the CPU can look at instructions in a machine code program and know exactly what to do without any further processing. However, machine code programs are unintelligible to humans; after all, a machine code program looks like a long list of numbers. Languages such as BASIC or Pascal, called *high level languages*, make more sense to people but cannot be executed directly by the CPU. The programs in these two languages need further processing before they can be executed by the CPU of the computer. There are two ways in which this can be done—*interpretation* and *compilation*.

Interpreted languages

In interpreted languages, like the BASIC on the BBC Micro or the popular GWBASIC for the IBM PC, the set of instructions that is written in the language is examined by a language interpreter and a series of machine code instructions are executed for each instruction in the high level language program. Thus a single high level instruction may give rise to several thousand machine code instructions, which are pre-programmed in to the language interpreter. If we had a simple loop in a program, like:

```
FOR I=1 TO 10
PRINT "HELLO WORLD"
NEXT I
```

then the statement *PRINT "HELLO WORLD"* would be reinterpreted ten times! This, combined with the fact that the machine code statements