Mathematical Models in Biology, An Introduction

Version 02.12.31

Elizabeth S. Allman[1]
John A. Rhodes[2]

1 Department of Mathematics and Statistics,
University of Southern Maine
2 Department of Mathematics,
Bates College

To J., R., and K.,
may reality live up to the model

Contents

Preface

Interactions between the mathematical and biological sciences have been increasing rapidly in recent years. Both traditional topics, such as population and disease modeling, and new ones, such as those in genomics arising from the accumulation of DNA sequence data, have made biomathematics an exciting field. The best predictions of numerous individuals and committees have suggested that the area will continue to be one of great growth.

We believe these interactions should be felt at the undergraduate level. Mathematics students gain from seeing some of the interesting areas open to them, and biology students benefit from learning how mathematical tools might help them pursue their own interests. The image of biology as a non-mathematical science, which persists among many college students, does a great disservice to those who hold it. This text is an attempt to present some substantive topics in mathematical biology at the early undergraduate level. We hope it may motivate some to continue their mathematical studies beyond the level traditional for biology students.

The students we had in mind while writing it have a strong interest in biological science, and a mathematical background sufficient to study calculus. We do not assume any training in calculus or beyond, as our focus on modeling through difference equations enables us to keep prerequisites minimal. Mathematical topics ordinarily spread through a variety of mathematics courses are introduced as needed for modeling or the analysis of models.

Despite this organization, we are aware that many students will have had calculus, and perhaps other mathematics courses. We therefore have not hesitated to include comments and problems (all clearly marked) that may benefit those with additional background. Our own classes using this text have included a number of students with extensive mathematical backgrounds, and they have found plenty to learn. Much of the material is also appealing to students in other disciplines who are simply curious. We believe the text can be used productively in many ways, for both classes and independent study, and at many levels.

Our writing style is intentionally informal. We have not tried to offer definitive coverage of any topic, but rather draw students into an interesting field. In particular, we often only introduce certain models and leave their analysis to exercises.

Though this would be an inefficient way to give encyclopedic exposure to topics, we hope it leads to deeper understanding and questioning.

Because computer experimentation with models can be so informative, we have supplemented the text with a number of MATLAB programs. MATLAB's simple interface, its widespread availability in both professional and student versions, and its emphasis on numerical rather than symbolic computation, have made it well-suited to our goals. We suggest appropriate MATLAB commands within problems, so that effort spent teaching its syntax should be minimal. While the computer is a tool students should use, it is by no means a focus of the text.

In addition to many exercises, a variety of projects are included. These propose a topic of study, and suggest ways to investigate it, but they are all at least partially open-ended. Not only does this allow students to work at different levels, it also is more true to the reality of mathematical and scientific work.

Throughout the text are questions marked with '► '. These are intended as gentle prods to prevent passive reading. Answers should be relatively clear after a little reflection, or the issue will be discussed in the text afterwards. If you find such nagging annoying, please feel free to ignore them.

There is more material in the text than could be covered in a semester, offering instructors many options. The topics of Chapters 1, 2, 3, and 7 are perhaps the most standard for mathematical biology courses, covering population and disease models, both linear and non-linear. Chapters 4 and 5 offer students an introduction to newer topics of molecular evolution and phylogenetic tree construction that are both appealing and useful. Chapter 6, on genetics, provides a glimpse of another area where mathematics and biology have long been intertwined. Chapter 8 and the appendix give a brief introduction to the basic tools of curve fitting and statistics.

In terms of logical development, mathematical topics are introduced as they are needed in addressing biological topics. Chapters 1 introduces the concepts of dynamic modeling through one-variable difference equations, including the key notions of equilibria, linearization, and stability. Chapter 2 motivates matrix algebra and eigenvector analysis through two-variable linear models. These chapters are a basis for all that follows.

An introduction to probability appears in two sections of Chapter 4, in order to model molecular evolution, and is then extended in Chapter 6 for genetics applications. Chapter 5, which has an algorithmic flavor different from the rest of the text, depends in part on the distance formulas derived in Chapter 4. Chapter 8's treatment of infectious disease models naturally depends on Chapter 3's introduction to models of interacting populations.

The development of this course began in 1994, with support from a Hughes Foundation grant to Bates College. Within a few years, brief versions of a few chapters written by the second author had evolved. The first author supplemented these with additional chapters, with support provided by the American Association of University Women. After many additional joint revisions, the course notes reached a critical mass where publishing them for others to use was no longer frightening. A Phillips grant from Bates and a professional leave from the University of Southern Maine aided the completion.

We thank our many colleagues, particularly those in the biological sciences, who aided us over the years. Seri Rudolph, Karen Rasmussen, and Melinda Harder

all helped outline the initial course, and Karen provided additional consultations until the end. Many students helped, both as assistants and classroom guinea pigs, testing problems and text and asking many questions. A few who deserve special mention are Sarah Baxter, Michelle Bradford, Brad Cranston, Jamie McDowell, Christopher Hallward, and Troy Shurtleff. We also thank Cheryl McCormick for informal consultations.

Despite our best intentions, errors are sure to have slipped by us. Please let us know of any you find.

Elizabeth Allman
eallman@maine.edu
John Rhodes
jrhodes@bates.edu
Portland, Maine
Turner, Maine

Note on MATLAB

Many of the exercises and projects refer to the computer package MATLAB. Learning enough of the basic MATLAB commands to use it as a high-powered calculator is both simple and worthwhile. When the text requires more advanced commands for exercises, examples are generally given within the statements of the problems. In this way, facility with the software can be built gradually.

MATLAB is in fact a complete programming language with excellent graphical capabilities. We have taken advantage of these features to provide a few programs making investigating the models in this text easier for the MATLAB beginner. Both exercises and projects refer to some of the programs (called `m`-files) or data files (called `mat`-files) below.

The `m`-files have been written to minimize necessary background knowledge of MATLAB syntax. To run most of the `m`-files below, say `onepop.m`, be sure it is in your current MATLAB directory or path and type `onepop`. You will then be asked a series of questions about models and parameters. The command `help onepop` also provides a brief description of the program's function. Since *m*-files are text files, they can be read and modified by anyone interested.

Some of the `m`-files define functions, which take arguments. For instance, a command like `compseq(seq1,seq2)` runs the program `compseq.m` to compare the two DNA sequences `seq1` and `seq2`. Typing `help compseq` prints an explanation of the syntax of such a function.

A `mat`-file contains data that may only be accessed from within MATLAB. To load such a file, say `seqdata.mat`, type `load seqdata`. The names of any new variables this creates can be seen by then typing `who`, while values stored in those variables can be seen by typing the variable name.

Some data files have been given in the form of *m*-files, so that supporting comments and explanations could be saved with the data. For these, running the *m*-file creates variables, just as loading a *mat*-file would. The comments can be read with any editor.

The MATLAB files made available with the text are:

- `aidsdata.m` — contains data from the CDC on AIDS cases in the United States
- `cobweb.m, cobweb2.m` — produces cobweb diagram movies for iterations of a one-population model; the first program leaves all web lines that are drawn, while the second gradually erases them
- `compseq.m` — compares two DNA sequences, producing a frequency table of the number of sites with each of the possible base combinations

- `distances.m` — computes Jukes-Cantor, Kimura 2-parameter, and log-det (paralinear) distances between all pairs in a collection of DNA sequences
- `distJC.m, distK2.m, distLD.m` — computes Jukes-Cantor, Kimura 2-parameter, or log-det (paralinear) distance for one pair of sequences described by a frequency array of sites with each base combination
- `flhivdata.m` — contains DNA sequences of the envelope gene for HIV from the 'Florida dentist case'
- `genemap.m` — simulates testcross data for a genetic mapping project, using either fly or mouse genes
- `genesim.m` — produces a time plot of allele frequency of a gene in a population of fixed size; relative fitness values for genotypes can be set to model natural selection
- `informative.m` — locates sites in aligned DNA sequences that are informative for the method of maximum parsimony
- `longterm.m` — draws a bifurcation diagram for a one-population model, showing long-term behavior as one parameter value varies
- `markovJC.m, markovK2.m` — produces a Markov matrix with of Jukes-Cantor or Kimura 2-parameter form with specified parameter values
- `mutate.m, mutatef.m` — simulates DNA sequence mutation according to a Markov model of base substitution; the second program is a function version of the first
- `nj.m` — performs the neighbor-joining algorithm to construct a tree from a distance array
- `onepop.m` — displays time plots of iterations of a one-population model
- `primatedata.m` — contains mitochondrial DNA sequences from 12 primates, as well as computed distances between them
- `seqdata.mat` — contains simulated DNA sequence data
- `seqgen.m` — generates DNA sequences with specified length and base distribution
- `sir.m` — displays iterations of an SIR epidemic model, including time and phase plane plots
- `twopop.m` — displays iterations of a two-population model, including time and phase plane plots

Of the above programs, `compseq`, `distances`, `distJC`, `distK2`, `distLD`, `informative`, `markovJC`, `markovK2`, `mutatef`, `nj`, and `seqgen` are functions requiring arguments.

Index