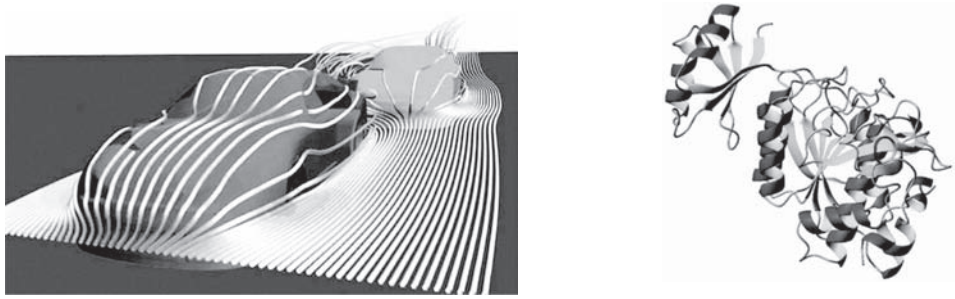
1 MATHEMATICAL MODELING

Numerical methods play an important role in modern science. Scientific exploration is often conducted on computers rather than laboratory equipment. While it is rarely meant to completely replace work in the scientific laboratory, computer simulation often complements this work.

For example, the aerodynamic simulation of two NASCAR autos pictured in figure 1.1(a) requires the numerical solution of *partial differential equations* (PDEs) that model the flow of air past the car. An auto body must be smooth and sleek, so it is often modeled using cubic (or higher order) *splines*. Similar computations are done in designing aircraft. We will study the numerical issues in using splines and solving PDEs in chapters 8 and 14, respectively.

Other examples occur in the field of mathematical biology, an active area of research in industry, government, and academia. Numerical algorithms play a crucial role in this work. For example, protein-folding models are often solved as large *optimization* problems. Protein arranges itself in such a way as to minimize energy—nature has no trouble finding the right arrangement, but it is not so easy for humans. The field of numerical optimization is an entire subject on its own, so it will not be covered in this book. The numerical methods described here, however, form the core of most optimization procedures.

Before studying issues in the analysis and implementation of efficient and accurate numerical methods, we first look briefly at the topic of mathematical modeling, which turns real-world problems into the sorts of mathematical equations that numerical analysts can tackle. The mathematical formulation usually represents only a *model* of the actual physical situation, and it is often important for the numerical analyst or computational scientist to know something about the origin of the model; in fact, numerical analysts sometimes work directly with scientists and engineers in devising the mathematical model. This interaction is important for a number of reasons. First, many algorithms do not produce the exact solution but only an approximate one. An understanding of the origin of the problem is necessary to determine what constitutes an acceptably good "approximate" solution: an error of a few centimeters might be acceptable in locating an enemy tank, but it would not be acceptable in locating a tumor for laser surgery! Second, even if the algorithm theoretically produces the exact solution, when implemented on a computer using finite-precision arithmetic, the results produced will most



(a) (b)

Figure 1.1. (a) A simulation of two NASCAR autos depicts the streamlines of air produced as a car drafts and is about to pass another. (Simulation performed with STAR-CCM+.) (b) Solving protein-folding models utilizes numerical optimization.

likely be inexact. Part of numerical analysis is the understanding of the impact of finite-precision computations on the accuracy of results. We will look more deeply at the issues in computing in finite precision in chapter 5.

In this chapter we present a variety of applications that involve numerical computation and come from the mathematical modeling of various processes.

1.1 MODELING IN COMPUTER ANIMATION

Many of the computer generated graphics that dominate the silver screen are produced with **dynamic simulation**; that is, a model is created, often using the laws of physics, and numerical methods are then used to compute the results of that model. In this section, we will look at the role of numerics in animation that appeared in the 2002 film *Star Wars: Episode II Attack of the Clones*. In particular, we will take a careful look at some of the special effects used to digitally create the character of Yoda, a Jedi master who first appeared as a puppet in the Star Wars saga in the 1980 film, *The Empire Strikes Back*. In the 2002 film, Yoda was digitally created, which required heavy use of numerical algorithms.

A key aspect of creating a digital Yoda involves producing believable movement of the character. The movement of Yoda's body is described using **key-frame animation**, in which a pose is specified at particular points in time and the computer automatically determines the poses in the intervening frames through interpolation. (We will discuss several interpolation techniques in chapter 8.) Animators have many controls over such movement, with the ability to specify, for instance, velocities and tangents of motion. While animators indicate the movement of Yoda's body, the computer must determine the resulting flow of his robe.

1.1.1 A Model Robe

Referring to figure 1.2, we see that the robe is represented with triangles and the motion of each vertex of the robe must be determined. Each vertex is

(a) (b) (c) (d)

Figure 1.2. Stages of simulation in the animation of the digitally created Yoda in the fight scene with Count Dooku in *Star Wars Episode II*. Two layers of Yoda's clothing, seen in (b) and (c), were computed separately. A process known as collision detection ensured that the inner layer of clothing did not intersect the outer robe and become visible. A simplified model of cloth illumination created the appearance of a real garment, producing the final rendering of the image in (d) [22]. (Courtesy of Lucasfilm Ltd. *Star Wars: Episode II - Attack of the Clones* ™ & © 2002 Lucasfilm Ltd. All rights reserved. Used under authorization. Unauthorized duplication is a violation of applicable law. Digital Work by Industrial Light & Magic.)

modeled as a particle, which in this context is a pointlike object that has mass, position, and velocity, and responds to forces, but has no size.

The motion of a particle is governed by Newton's second law, which is expressed mathematically by the equation

$$\mathbf{F} = m\mathbf{a} = m(d^2\mathbf{y}/dt^2), \tag{1.1}$$

where $\mathbf{y}$ is a distance function of time t. Note that the equations involve vector-valued functions since our computations are performed in three dimensions. Since a particle has mass ($m \neq 0$), equation (1.1) can be rewritten as the second-order ordinary differential equation (ODE)

$$\frac{d^2\mathbf{y}}{dt^2} = \frac{\mathbf{F}}{m}. \tag{1.2}$$

This ODE is part of an initial value problem since the state of the particle at some initial time is given. In the case of a movie, this is where the scene (which may be several seconds or a fraction of a second in duration) begins.

To keep the shape of the robe, pairs of neighboring particles are attached to each other using a spring force. Hooke's law states that a spring exerts a force F_s that is proportional to its displacement from its rest length x_0. This is expressed mathematically as

$$F_s = -k(x - x_0),$$

where x denotes the current position of the spring and k is the spring constant. For simplicity, we have stated the one-dimensional formulation of Hooke's law, but to model the Jedi's robe a three-dimensional version is used.

Many other forces are computed to animate Yoda's robe, including gravity, wind forces, collision forces, friction, and even completely made-up forces that are invented solely to achieve the motion that the director requests. In the

simplest of cases, an *analytic solution* of the model may exist. In computer animation, however, the forces acting on the particle constantly change and finding an analytic solution for each frame—if even possible—would be impractical. Instead, numerical methods are used to find approximate solutions by simulating the motion of the particles over discrete time steps. There is a large body of literature on the numerical solution of initial value problems for ODEs, and some of the methods will be covered in chapter 11.

BENDING THE LAWS OF PHYSICS

When simulating the motion of Jedi robes in *Star Wars Episode II*, animators discovered that if the stunts were performed in reality, the clothing would be ripped apart by such accelerations of the motion. To solve the problem, custom "protection" effects were added to the simulations, which dampened this acceleration. In the end, the clothes on the digital actors were less distorted by their superhuman motion. To the left we see a digital double of Obi-Wan Kenobi (played by Ewan McGregor), performing a stunt too dangerous for a live actor, in *Star Wars Episode II*. Note that the hairs, like the clothing, were simulated as strips of particles connected by springs [22]. (Courtesy of Lucasfilm Ltd. *Star Wars: Episode II - Attack of the Clones* ™ & © 2002 Lucasfilm Ltd. All rights reserved. Used under authorization. Unauthorized duplication is a violation of applicable law. Digital Work by Industrial Light & Magic.)

A goal of movie animation is creating convincing simulations. Accuracy, which is a leading goal of scientific simulation, may conflict with this goal. As such, numerical simulations are used for different goals by scientists and by the entertainment industry, but many of the mathematical tools and strategies are common. Let us now turn our attention to a simulation in science.

1.2 MODELING IN PHYSICS: RADIATION TRANSPORT

The transport of radiation can be described stochastically and modeled using Monte Carlo simulation. Monte Carlo methods will be described in chapter 3. Radiation transport also can be modeled by an integro-differential equation, the Boltzmann transport equation. This is a PDE that also involves an integral

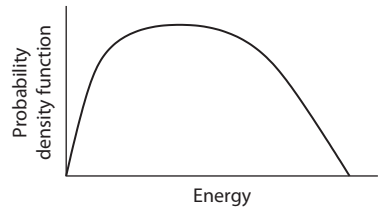


Figure 1.3. An example distribution of the energy of photons.

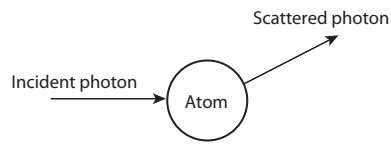


Figure 1.4. A collision between a photon and an atom.

term. In chapter 14 we discuss the numerical solution of PDEs, while chapter 10 describes methods of numerical integration. A combination of these ideas, together with iterative techniques for solving large linear systems (section 12.2), is used to approximate solutions to the Boltzmann transport equation.

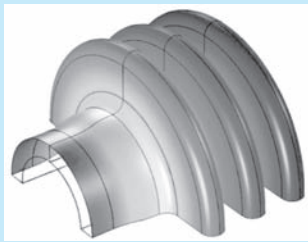
What radiation dose does your body receive from a dental X-ray examination? You probably recall that a heavy vest is usually placed over you during the exam and that the dental assistant leaves the room while the X-ray machine is on. The purpose of the covering is to absorb radiation. How can the transport of X-ray photons be modeled mathematically in order to aid in the design of such protective materials and to verify their effectiveness? The photons are produced by an electron beam that is turned on and off as X-rays are needed. The energy and direction of travel of any individual photon cannot be predicted, but the overall distribution of energy and direction of the X-rays can be approximated.

The independent variables in the system are energy, position, direction, and time of production of the photons, and each of these variables can be thought of as random but obeying a certain distribution. The energy of the photons might be distributed as shown in figure 1.3, for example.

It is assumed that particles move in straight lines until they enter matter, where, with a certain probability (depending on the characteristics of the material), they collide with an atom. The collision usually involves an exchange of energy with an electron in the atom, after which the photon emerges with reduced energy and an altered direction of travel. The photon may even give up all of its energy to the electron, in which case it is considered to be absorbed by the atom.

The probabilities of each of these events must be known in order to simulate the situation. These probabilities are deduced from theoretical and measured properties of X-rays and of various materials that might be used as shields. One can then run a computer simulation, often with millions of photons, each following a random path determined by these probability distributions. The history of each photon is followed until it is either absorbed (or loses so much energy that it effectively can be ignored) or travels outside the system to a point

from which it will not return. Average results are then taken to determine what dose of radiation is received at a particular location.



STUDYING SEMICONDUCTORS

Studying the behavior of electrons in semiconductor materials requires solving the Boltzmann transport equation, which involves complicated integrals. Both deterministic methods and Monte Carlo methods are sometimes used in this case. The picture to the left is a finite element discretization used in a deterministic model for radiation transport problems. (Image reproduced with the kind permission of the Applied Modelling and Computational Group at Imperial College London and EDF Energy. All other rights of the copyright owners are reserved.)

1.3 MODELING IN SPORTS

In FIFA World Cup soccer matches, soccer balls curve and swerve through the air, in the players' attempts to confuse goalkeepers and send the ball sailing to the back of the net. World class soccer players such as Brazil's Roberto Carlos, Germany's Michael Ballack and England's David Beckham have perfected "bending" the ball from a free kick.

According to Computational Fluid Dynamics (CFD) research by the University of Sheffield's Sports Engineering Research Group and Fluent Europe, the shape and surface of the soccer ball, as well as its initial orientation, play a fundamental role in the ball's trajectory through the air. In particular, such CFD research has increased the understanding of the "knuckleball" effect sometimes used to confuse an opposing goalkeeper who stands as the last line of defense. To obtain such results, a soccer ball was digitized down to its stitching as seen in figure 1.5. Note the refinement near the seams, which is required in order to properly model the boundary layer.

Some free kicks in soccer have an initial velocity of almost 70 mph. Wind tunnel experiments demonstrate that a soccer ball moves from laminar to turbulent flow at speeds between 20 and 30 mph, depending on the ball's surface structure and texture.

The techniques developed in Sheffield facilitated detailed analysis of the memorable goal by David Beckham of England versus Greece during the World Cup Qualifiers in 2001. In a sense, Beckham's kick applied sophisticated physics. While the CFD simulations at Sheffield can accurately model turbulent flow only when it is averaged over time, and so cannot yet give realistic trajectories in all cases, such research could affect soccer players from beginner

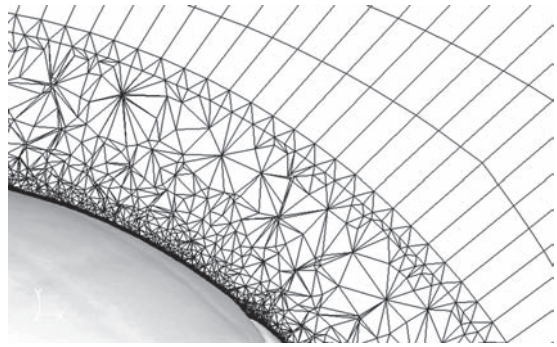
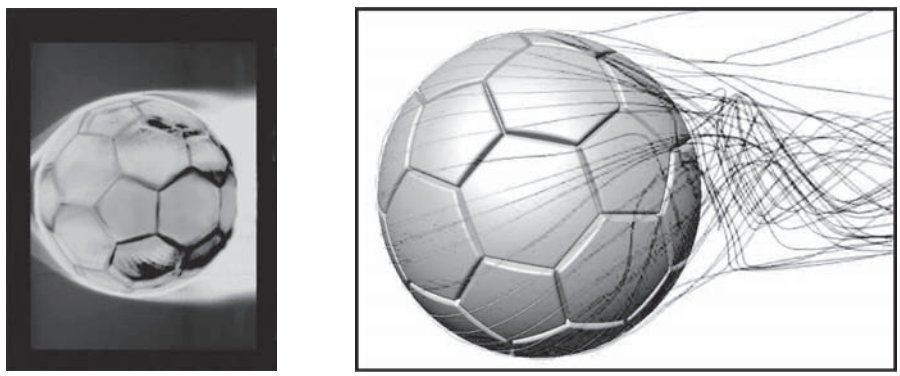


Figure 1.5. An important step in CFD simulations at the University of Sheffield is capturing the geometry of a soccer ball with a three-dimensional noncontact laser scanner. The figure shows part of a soccer ball mesh with approximately 9 million cells. (Courtesy of the University of Sheffield and Ansys UK.)



(a) (b)

Figure 1.6. (a) Wind tunnel smoke test of a nonspinning soccer ball. (b) CFD simulation showing wake-flow path lines of a nonspinning soccer ball, air speed of 27 mph. (Courtesy of the University of Sheffield and Ansys UK.)

to professional. For instance, ball manufacturers could exploit such work to produce a more consistent or interesting ball that could be tailored to the needs and levels of players. Such work could also impact the training of players. For more information, see, for instance [6] or [7].

To this end, there is a simulation program called Soccer Sim developed at the University of Sheffield. The program predicts the flight of a ball given input conditions, which can be acquired from CFD and wind tunnel tests, as well as from high-speed videoing of players' kicks. The software can then be used to compare the trajectory of a ball given varying initial orientations of the ball or different spins induced by the kick. Moreover, the trajectory can be compared for different soccer balls.

Note that this application, like that in section 1.1, involves the solution of differential equations. The next application that we discuss involves a discrete phenomenon.

Figure 1.7. High-speed airflow path lines colored by local velocity over the 2006 Teamgeist soccer ball. (Courtesy of the University of Sheffield and Ansys UK.)

1.4 ECOLOGICAL MODELS

Computational biology is a growing field of application for numerical methods. In this section, we explore a simplified example from ecology.

Suppose we wish to study the population of a certain species of bird. These birds are born in the spring and live at most 3 years. We will keep track of the population just before breeding, when there will be three classes of birds, based on age: Age 0 (born the previous spring), Age 1, and Age 2. Let $v_0^{(n)}$, $v_1^{(n)}$ and $v_2^{(n)}$ represent the number of females in each age class in Year n. To model population changes we need to know:

- Survival rates. Suppose 20% of Age 0 birds survive to the next spring, and 50% of Age 1 birds survive to become Age 2.
- Fecundity rates. Suppose females that are 1 year old produce a clutch of a certain size, of which 3 females are expected to survive to the next breeding season. Females that are 2 years old lay more eggs and suppose that of these, 6 females are expected to survive to the next spring.

Then we have the following model of the number of females in each age class:

$$\begin{aligned} v_0^{(n+1)} &= 3v_1^{(n)} + 6v_2^{(n)}, \\ v_1^{(n+1)} &= 0.2v_0^{(n)}, \\ v_2^{(n+1)} &= 0.5v_1^{(n)}. \end{aligned}$$

In matrix–vector form,

$$\begin{aligned} \mathbf{v}^{(n+1)} &= A\mathbf{v}^{(n)} \\ &= \begin{pmatrix} 0 & 3 & 6 \\ 0.2 & 0 & 0 \\ 0 & 0.5 & 0 \end{pmatrix} \begin{pmatrix} v_0^{(n)} \\ v_1^{(n)} \\ v_2^{(n)} \end{pmatrix}. \end{aligned} \tag{1.3}$$

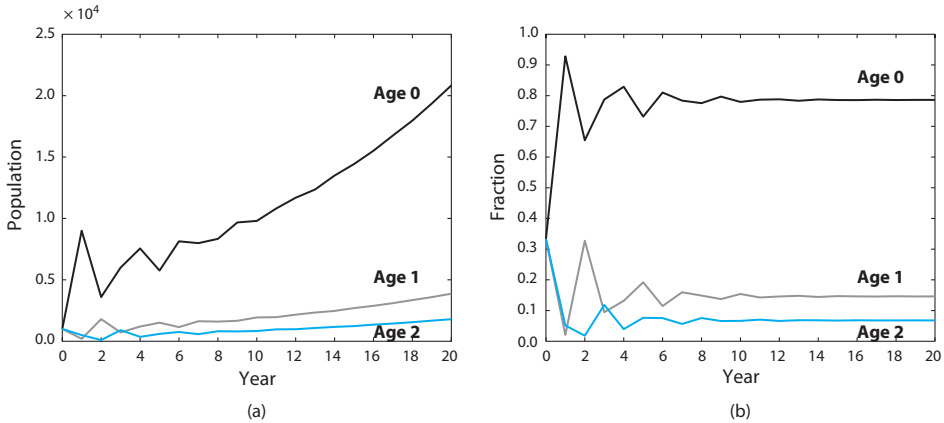


Figure 1.8. Bird population with $a_{32} = 0.5$ and initial data $\mathbf{v}^{(0)} = (1000, 1000, 1000)^T$.

The matrix A in (1.3) can be used to predict future populations from year to year and expected long-term behavior. This type of matrix, reflecting survival rates and fecundities, is called a **Leslie matrix**.

Suppose we start with a population of 3000 females, 1000 of each age. Using (1.3), we find

$$\mathbf{v}^{(0)} = \begin{pmatrix} 1000 \\ 1000 \\ 1000 \end{pmatrix}, \qquad \mathbf{v}^{(1)} = A\mathbf{v}^{(0)} = \begin{pmatrix} 0 & 3 & 6 \\ 0.2 & 0 & 0 \\ 0 & 0.5 & 0 \end{pmatrix} \mathbf{v}^{(0)} = \begin{pmatrix} 9000 \\ 200 \\ 500 \end{pmatrix},$$

$$\mathbf{v}^{(2)} = A\mathbf{v}^{(1)} = \begin{pmatrix} 3600 \\ 1800 \\ 100 \end{pmatrix}, \quad \text{and} \quad \mathbf{v}^{(3)} = A\mathbf{v}^{(2)} = \begin{pmatrix} 6000 \\ 720 \\ 900 \end{pmatrix}.$$

Plotting the population in each age group as a function of the year produces the graph in figure 1.8(a). Clearly the population grows exponentially. Figure 1.8(b) shows the proportion of the population in each age class as a function of the year. Note how the proportion of birds in each age class settles into a steady state.

To be more quantitative, we might look, for instance, at the population vectors $\mathbf{v}^{(20)}$ and $\mathbf{v}^{(19)}$. We find that $\mathbf{v}^{(20)} = (20833, 3873, 1797)^T$, which indicates that in year 20 the fraction of birds in each age class is $(0.7861, 0.1461, 0.068)^T$. Looking at the difference between $\mathbf{v}^{(20)}$ and $\mathbf{v}^{(19)}$, we find that the total population grows by a factor of 1.0760 between year 19 and year 20.

Note that $\mathbf{v}^{(n)} = A\mathbf{v}^{(n-1)} = A^2\mathbf{v}^{(n-2)} = \ldots = A^n\mathbf{v}^{(0)}$. In chapter 12, we will see how this observation indicates that the asymptotic behavior of this system (the behavior after a long time period) can be predicted from the dominant eigenvalue (the one of largest absolute value) and associated eigenvector of A. The eigenvalues of A can be computed to be 1.0759, $-0.5380 + 0.5179i$, and $-0.5380 - 0.5179i$ (where $i = \sqrt{-1}$). The largest magnitude of an eigenvalue is 1.0759, and the associated eigenvector, scaled so that its entries sum to 1, is $\hat{\mathbf{v}} = (0.7860, 0.1461, 0.0679)$. The largest magnitude of an eigenvalue

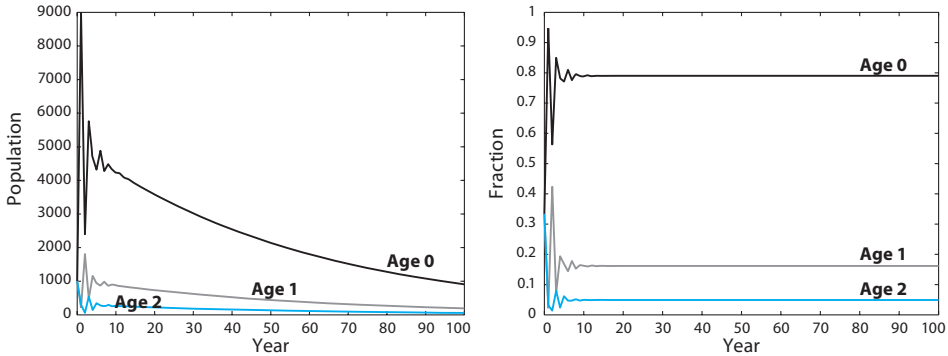


Figure 1.9. Bird population with $a_{32} = 0.3$ and initial data $\mathbf{v}^{(0)} = (1000, 1000, 1000)^T$.

determines the rate of growth of the population after a long period of time, and the entries in $\hat{\mathbf{v}}$ give the fractions of birds in each class, as these fractions approach a steady state. While the eigenvalues and eigenvectors of a 3 by 3 matrix such as A can be determined analytically, we will learn in chapter 12 about methods to numerically approximate eigenvalues and eigenvectors of much larger matrices.

Harvesting Strategies. We can use this model to investigate questions such as the following. Suppose we want to harvest a certain fraction of this population, either to control the exponential growth or to use it as a food source (or both). How much should we harvest in order to control the growth without causing extinction?

Harvesting will decrease the survival rates. We might wish to decrease these rates to a point where the dominant eigenvalue is very close to 1 in absolute value, since this would mean that the total population, after a period of time, would remain fixed. If we decrease the survival rates too much, then all eigenvalues will be less than 1 in magnitude and the population size will decrease exponentially to extinction.

For example, suppose we harvest 20% of the 1-year-olds, so that the survival rate falls from 0.5 to 0.3. In this case, entry a_{32} in matrix A changes from 0.5 to 0.3, and it turns out that the largest magnitude eigenvalue decreases to 0.9830. A simulation with these parameters yields the graphs in figure 1.9.

Note that we now have exponential decay of the population and eventual extinction. The decrease in population is gradual, however, and over the 100-year time span shown here it decreases only from 3000 to 1148. Asymptotically (i.e., after a long period of time), the total population will decrease each year to 0.9830 times that of the previous year. Hence if the population is 1148 after 100 years of this harvesting strategy, the number of additional years before the population drops below 1 (i.e., the species becomes extinct), might be estimated by the value of k that satisfies $1148(0.9830)^k = 1$, which is $k = -\log(1148)/\log(0.9830) \approx 411$ years.

One could try to find the matrix element a_{32} that results in the largest magnitude eigenvalue of A being exactly 1, and while this is an interesting

mathematical problem, from the point of view of population control, it would likely be an exercise in futility. Clearly, this model is highly simplified, and even if the assumptions about survival rates and fecundity rates held initially, they might well change over time. Populations would need to be monitored frequently to adjust a particular harvesting strategy to the changing environment.

1.5 MODELING A WEB SURFER AND GOOGLE

Submitting a query to a search engine is a common method of information retrieval. Companies compete to be listed high in the rankings returned by a search engine. In fact, some companies' business is to help raise the rankings of a paying customer's web page. This is done by exploiting knowledge of the algorithms used by search engines. While a certain amount of information about search engine algorithms is publicly available, there is a certain amount that is proprietary. This section discusses how one can rank web pages based on content and is introductory in nature. An interested reader is encouraged to research the literature on search engine analysis, which is an ever-growing field. Here we will consider a simple vector space model for performing a search. This method does not take into account the hyperlink structure of the World Wide Web, and so the rankings from such a model might be aggregated with the results of an algorithm that does consider the Web's hyperlink structure. Google's PageRank algorithm is such a method and will be looked at briefly in this section and in more detail in sections 3.4 and 12.1.5.

1.5.1 The Vector Space Model

The vector space model consists of two main parts: a list of documents and a dictionary. The list of documents consists of those documents on which searches are to be conducted, and the dictionary of terms is a database of keywords. While the dictionary of terms could be all the terms in every document, this may not be desirable or computationally practical. Words not in the dictionary return empty searches.

With a list of n documents and m keywords, the vector space model constructs an m by n *document matrix* A with

$$a_{ij} = \begin{cases} 1 & \text{if document } j \text{ is relevant to term } i, \\ 0 & \text{otherwise.} \end{cases} \tag{1.4}$$

When a query is issued, a query vector $\mathbf{q} = (q_1, \ldots, q_m)^T$ is then formed, with $q_i = 1$ if the query includes term i and $q_i = 0$ otherwise. The "closeness" of the query to each document is then measured by the cosine of the angle between the query vector $\mathbf{q}$ and the column of A representing that document. If $\mathbf{a}_j$ denotes the jth column of A (and if it is not entirely zero due to document j containing no keywords), then the angle θ_j between $\mathbf{q}$ and $\mathbf{a}_j$ satisfies

$$\cos(\theta_j) = \frac{\mathbf{a}_j^T \mathbf{q}}{\|\mathbf{a}_j\|_2 \|\mathbf{q}\|_2}. \tag{1.5}$$

TABLE 1.1
The dictionary of terms and documents used in a three-dimensional example.

Dictionary	Documents	
electric	I	the art of war
fencing	II	the fencing master
foil	III	fencing techniques of foil, epee, and saber
	IV	hot tips on building electric fencing

Since a larger value of the cosine implies a smaller angle between the vectors, documents are ranked in order of relevance to the query by arranging the cosines in descending order.

A major difficulty in forming the document matrix is determining whether a document is relevant to a term. There are a variety of ways of doing this. For instance, every document can be searched for each keyword. A document is deemed relevant to those keywords that are contained within it. This requires considerable computational expense when one faces a large number of documents and keywords. An alternative that is employed by some search engines is to read and perform analysis on only a portion of a document's text. For example, Google and Yahoo! pull only around 100K and 500K bytes, respectively, of web page text [12]. Other choices that must be made include whether to require exact matching of terms or to allow synonyms, and whether or not to count word order. For example, do we treat queries of `boat show` and `show boat` as the same or different? All of these choices impact which documents are deemed most relevant [36].

To illustrate the method, we give a simple example, in which a document is deemed relevant if its title contains the keyword exactly.

Searching in a Tiny Space

Suppose that our dictionary contains three keywords—"electric", "fencing", and "foil"—and that there are four documents entitled "the art of war", "the fencing master", "fencing techniques of foil, epee, and saber", and "hot tips on building electric fencing". We have written the document titles and dictionary terms in lower case to avoid questions about matching in the presence of such a factor. The information is listed in table 1.1 for easy reference.

To form the document matrix A—where rows 1, 2, and 3 correspond to the terms "electric", "fencing", and "foil", respectively, while columns 1, 2, 3 and 4 correspond to documents I, II, III, and IV, respectively—we use (1.4) to obtain

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

Figure 1.10 depicts the columns of A (each except the first normalized to have length 1) as vectors in three-space.

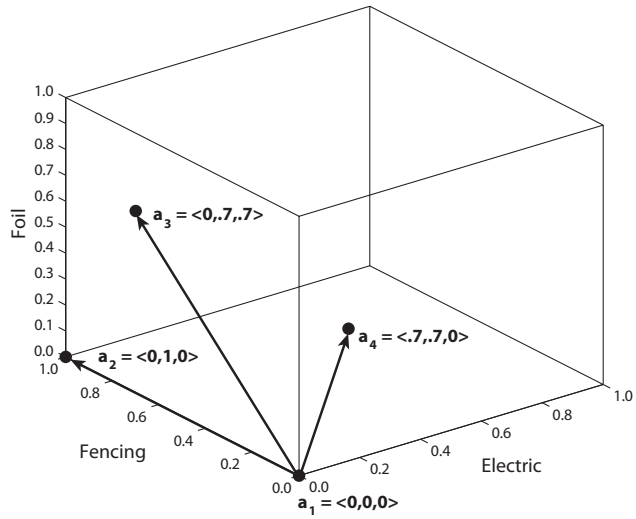


Figure 1.10. Visualization of vector space search in a three-dimensional example.

Suppose our query is `fencing`. Then the query vector is $\mathbf{q} = (0, 1, 0)^T$, and we can see by inspection that it is identical to column 2 of A. Thus, document II would be deemed most relevant to this query. To determine the rankings of the remaining documents, we compute the cosine of the angle between $\mathbf{q}$ and every other nonzero column of A using (1.5) to find

$$\cos(\theta_3) = \frac{\mathbf{a}_3^T\mathbf{q}}{\|\mathbf{a}_3\|_2\|\mathbf{q}\|_2} = \frac{1}{\sqrt{2}},$$

$$\cos(\theta_4) = \frac{\mathbf{a}_4^T\mathbf{q}}{\|\mathbf{a}_4\|_2\|\mathbf{q}\|_2} = \frac{1}{\sqrt{2}}.$$

Thus documents III and IV would be ranked equally, behind document II.

1.5.2 Google's PageRank

Real-world search engines deal with some of the largest mathematical and computer science problems in today's computational world. Interestingly, "Google" is a play on the word "googol", the number 10^{100}, reflecting the company's goal of organizing all information on the World Wide Web.

When you submit a query, such as `numerical analysis`, to Google, how does the search engine distinguish between the web page listed first and the one listed, say, 100th? There are various factors that play into this decision, one of which is the web page's relevance to your query, as discussed previously. Another important component, however, is the "quality" or "importance" of the page; "important" pages, or pages that are pointed to by many other pages, are more likely to be of interest to you. An algorithm called PageRank is used to measure the importance of a web page. This algorithm relies on a model of web-surfing behavior.

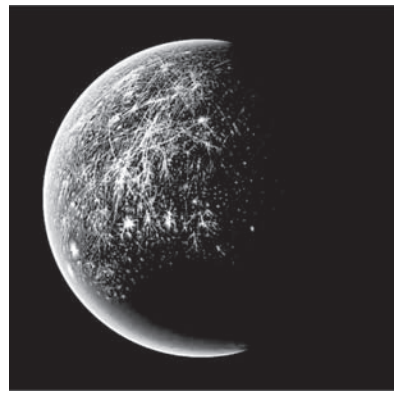


Figure 1.11. A representation of the graph of the Internet created by David F. Gleich, based on an image and data from the OPTE project.

As with any model of reality, Google's model of web-surfing behavior is an approximation. An important feature of PageRank is its assumption about the percentage of time that a surfer follows a link on the current web page. The exact assumptions that are made are proprietary, but it is believed that the PageRank algorithm used by Google assumes that a surfer follows a link on a web page about 85% of the time, with any of the links being equally likely. The other 15% of the time the surfer will enter the URL for another web page, possibly the same one that is currently being visited.

These assumptions about surfing behavior affect not only the accuracy of the model but also the efficiency of numerical techniques used to solve the model. Keep in mind that Google indexes billions of web pages, making this one of the largest computational problems ever solved! In chapter 12 we discuss more about the numerical methods used to tackle this problem.

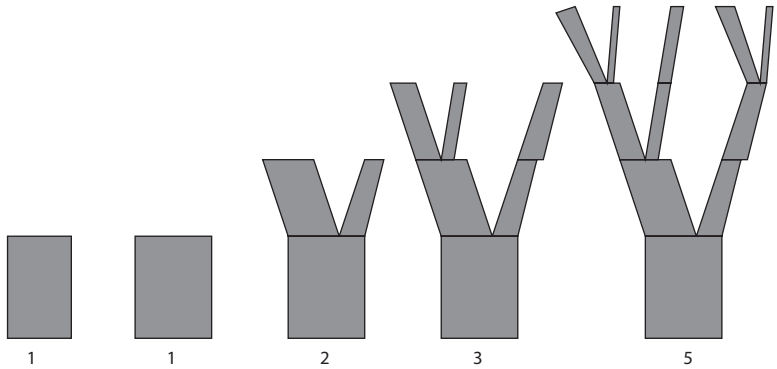
1.6 CHAPTER 1 EXERCISES

1. **The Fibonacci numbers and nature.** The Fibonacci numbers are defined by $F_1 = 1$, $F_2 = 1$, $F_3 = F_2 + F_1 = 2$, $F_4 = F_3 + F_2 = 3$, etc. In general, $F_{j+1} = F_j + F_{j-1}$, $j = 2, 3, \ldots$. Write down F_5, F_6, and F_7.

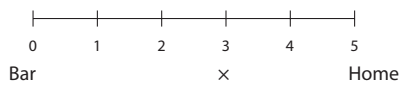
   It is often observed that the number of petals on a flower or the number of branches on a tree is a Fibonacci number. For example, most daisies have either 34, 55, or 89 petals, and these are the 9th, 10th, and 11th Fibonacci numbers. The reason four-leaf clovers are so rare is that 4 is not a Fibonacci number.

   To see the reason for this, consider the following model of branch growth in trees. We start with the main trunk ($F_1 = 1$), which spends one season growing ($F_2 = 1$), and then after another season, develops two branches ($F_3 = 2$)—a major one and a minor one. After the next season, the major branch develops two branches—a major one and a minor one—while the minor branch grows into a major one, ready to divide in the following season. At this point there are $F_4 = 3$ branches—two major ones and one

minor one. At the end of the next season, the two major branches divide, producing two major branches and two minor ones, while the minor one from the previous season grows into a major one. Thus at this point there are $F_5 = 5$ branches, with $F_4 = 3$ of these being major branches ready to divide during the next season. Explain why the Fibonacci numbers arise from this model of branch growth.



2. **Drunkard's walk.** A drunkard starts at position x in the diagram below and with each step moves right one space with probability .5 and left one space with probability .5. If he reaches the bar, he stays there, drinking himself into oblivion. If he reaches home, he goes to bed and stays there. You wish to know the probability $p(x)$ that he reaches home before reaching the bar.

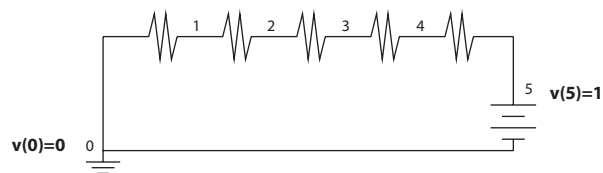


This is a typical Markov chain problem that will be discussed later in the text. Note that $p(0) = 0$ and $p(5) = 1$, since the drunk does not leave either the bar or his home. For $x = 1, 2, 3, 4$, $p(x) = .5\,p(x-1) + .5\,p(x+1)$, since he moves left with probability .5 and right with probability .5.

(a) Let the drunk's starting position be $x = 3$. What are the possible positions that he could be in after one step, and what are the probabilities of each? How about after two steps?
(b) For which initial positions x would you expect him to reach the bar first and for which would you expect him to reach home first, and why?

This is a typical *random walk* problem, and while it is posed as a silly story, it has real physical applications.

Consider an electrical network with equal resistors in series and a unit voltage across the ends.



Voltages $v(x)$ will be established at points $x = 0, 1, 2, 3, 4, 5$. We have grounded the point $x = 0$ so that $v(0) = 0$, and there is no resistor between the source and point $x = 5$, so that $v(5) = 1$. By Kirchoff's laws, the current flowing into x must be equal to the current flowing out. By Ohm's law, if points x and y are separated by a resistor of strength R, then the current i_{xy} that flows from x to y is

$$i_{xy} = \frac{v(x) - v(y)}{R}.$$

Thus for $x = 1, 2, 3, 4$, we have

$$\frac{v(x-1) - v(x)}{R} = \frac{v(x) - v(x+1)}{R}.$$

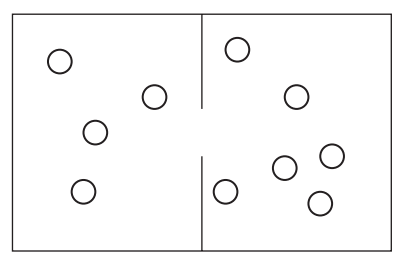
Multiplying by R and combining like terms we see that $v(x) = .5v(x - 1) + .5v(x + 1)$. This is exactly the same formula (with the same boundary conditions) that we found for $p(x)$ in the drunkard's walk problem.

Can you think of other situations that might be modeled in this same way? The behavior of a stock price perhaps? Suppose you generalize by allowing different probabilities of the drunkard moving right or left: say, the probability of moving right is .6 while that of moving left is .4. What generalization of the resistor problem would this correspond to? [Hint: Consider resistors of different strengths.]

3. **Ehrenfests' urn.** Consider two urns, with N balls distributed between them. At each unit of time, you take a ball from one urn and move it to the other, with the probability of choosing each ball being equal, $1/N$. Thus, the probability of choosing a ball from a given urn is proportional to the number of balls in that urn. Let $X(t)$ denote the number of balls in the left urn at time t, and suppose that $X(0) = 0$. Then $X(1) = 1$, since a ball must be drawn from the right urn and moved to the left one.

   (a) Let $N = 100$. What are the possible values for $X(2)$, $X(3)$, and $X(4)$, and what is the probability of each?

   (a) If this process were carried out for a very long time, what do you think would be the most frequently occurring value of $X(t)$?



This model was introduced in the early 1900s by Paul Ehrenfest and Tatiana Ehrenfest-Afanassjewa to describe the diffusion of gas molecules through a permeable membrane. See, for example, http://en.wikipedia.org/wiki/Ehrenfest_model for a discussion of its relation to the second law

of thermodynamics. In chapter 3 we will discuss Monte Carlo simulation of physical processes using models such as this one. Can you think of other situations that might be modeled by such a process? In addition to physical or chemical processes, you might consider, for example, financial decisions or social interactions.

4. Each year undergraduates participate in the *Mathematical Contest in Modeling (MCM)*. See www.mcm.org. Following is an example of the sort of modeling problems that they tackle:

> An ornamental fountain in a large open plaza surrounded by buildings squirts water high into the air. On gusty days, the wind blows spray from the fountain onto passersby. The water-flow from the fountain is controlled by a mechanism linked to an anemometer (which measures wind speed and direction) located on top of an adjacent building. The objective of this control is to provide passersby with an acceptable balance between an attractive spectacle and a soaking: The harder the wind blows, the lower the water volume and height to which the water is squirted, hence the less spray falls outside the pool area.
>
> Your task is to devise an algorithm which uses data provided by the anemometer to adjust the water-flow from the fountain as the wind conditions change.

   Think about how you might create a mathematical model for this problem and compare your ideas with some of the students' solutions that can be found in: *UMAP: Journal of Undergraduate Mathematics and its Applications*. 2002. 23(3):187–271.

5. Consider the following dictionary of keywords:

   chocolate, ice cream, sprinkles,

   and the following list of documents:

   D1. I eat only the chocolate icing off the cake
   D2. I like chocolate and vanilla ice cream
   D3. Children like chocolate cake with sprinkles
   D4. May I have another scoop of ice cream if you hold both the sprinkles and chocolate sauce

   Form the document matrix A, and, using the vector space model, rank the documents D1, D2, D3, and D4 according to their relevance to the query: `chocolate, ice cream`.

6. When you submit a query to a search engine, an ordered list of web pages is returned. The pages are ranked by their relevance to your query and also by the quality of the page. Consider the small network of web pages in figure 1.12. We will assume this is the set of web pages indexed by our search engine. Each vertex in the graph is a web page. A directed link is drawn from web page i to web page j if web page i has a link to web page j. So, we can see, for example, that web page 1 links to web page 4. The PageRank algorithm, as proposed by Larry Page and Sergey Brin [18], assumes that a surfer follows a link on a web page 85% of the time,

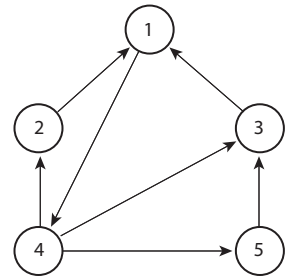


Figure 1.12. A small network of web pages.

with any of the links being equally likely. The other 15% of the time the surfer will enter the URL for another web page, possibly the same one that is currently being visited, again with all pages being equally likely. Let $X_i(t)$ denote the probability of being at web page i after the surfer takes t steps through the network. We will assume the surfer starts at web page 1, so that $X_1(0) = 1$ and $X_i(0) = 0$ for $i = 2, 3, 4, 5$.

(a) Find $X_i(1)$ for $1 \leq i \leq 5$.
(b) Find $X_i(2)$ for $1 \leq i \leq 5$.

As a measure of the quality of a page, PageRank approximates $\lim_{t\to\infty} X_i(t)$ for all i.