3 Scalar, Stochastic, Discrete Dynamic Systems

The sand-hill crane example in the textbook describes a population of birds in year $n$ by the first-order, deterministic recurrence equation

$$y(n) = Ry(n - 1) + u(n)$$

where

$$R = 1 + r = 1 + b - d.$$  

In this expression, the growth rate $r$ is the difference between the birth rate $b$ and the death rate $d$. The parameters $b$ and $d$ are numbers between zero and one, so that $R$ is between zero and two. If the recurrence above describes something other than a population (say, an amount of money), then negative values of $R$ are meaningful as well (debit versus credit), and we saw on page 29 of these notes that different values of $R$ yield a relatively rich variety of possible responses even when the input $u(n)$ to the recurrence is zero.

However, the output $y(n)$ in the sand-hill crane example had to be interpreted in any case as the average number of birds, rather than the actual number, if nothing else because when $R$ is a real number $y(n)$ is not necessarily an integer number. A deeper reason for thinking of $y(n)$ as an average is that the exact number of cranes is hard or impossible to predict. Consider the experiment of observing a population of sand-hill cranes over, say, $N = 10$ years from an initial population $y_0$. Repeating the experiment would entail reproducing the same environmental conditions, restart the population at $y_0$, and observe its evolution over another ten years. Different experiments are bound to produce different time sequences $y(0), y(1), \ldots$, although $y(0)$ in particular would be the same in all experiments. Discrepancies are caused by uncontrolled variations in the environment, by the health and genetic makeup of the specific initial population, and by other unpredictable factors such as whether a particular alligator was or was not able to capture and kill a particular bird on day 37 of year 4.

In summary, a bird population is a stochastic quantity, that is, a quantity whose exact variations defy detailed modeling, and are rather described in an aggregate sense. Another example of a stochastic quantity is the outcome of the roll of a die. A classical, mechanistic view of physics would have posited that it is possible in principle to know enough about the circumstances in which a die is cast to predict the outcome. In practice this is unrealistic, and in more modern views of physics utterly impossible. An aggregate description is more feasible, and would state that the probability of any of the six possible outcomes is the same for a fair die. In one interpretation, this means that if a fair die is rolled $K$ times, then each of the possible outcome values 1 through 6 is likely\(^{18}\) to occur about $K/6$ times, and the approximation improves indefinitely as $K$ increases.

A less detailed description would state that the average outcome of the roll of a die is 3.5. In one interpretation, this means that if a fair die is rolled $K$ times with outcomes $o_1, \ldots, o_K$, then the quantity

$$\frac{1}{K} \sum_{k=1}^{K} o_k$$

\(^{18}\)The astute reader will have noticed that the expression “is likely” reeks itself of probability. This observation is correct, and can be made precise.
is likely to become closer and closer to 3.5 as \( K \) increases.

Similarly, a growth coefficient of \( R \) could be interpreted by stating that if the experiment mentioned above were repeated \( K \) times, and the ratios, say, \( R_k = [y(1)/y(0)]_{\text{experiment } k} \) were computed from empirical observations over all experiments, then one would obtain

\[
\frac{1}{K} \sum_{k=1}^{K} R_k \approx R
\]

for a large enough \( K \).\(^{19}\)

The average outcome \( y(n) \) from a recurrence based on the average growth coefficient \( R \) is incomplete information, just as the statement that a roll of a die yields 3.5 on average is incomplete. Much more detailed information could be obtained if the recurrence under study were to also model the stochastic variations from experiment to experiment. This greater amount of information is both a curse and a blessing: It is a curse in that running such a recurrence once through a sequence of years would only provide information about that particular sequence, and would therefore be of limited predictive value. More detailed information is a blessing in that a recurrence can be run multiple times through a sequence of years, and by doing so one can compute aggregate information that includes but is not limited to the average behavior of the system.

A recurrence that includes stochastic behavior is called a stochastic dynamic system. Such a recurrence requires some mechanism for generating random outcomes. Even once such a mechanism is available, there is a wide choice of options for how to inject randomness into a recurrence. We start from a conceptually straightforward method next, for motivation and intuition. Some preliminaries on probability theory follow, and a more quantitative treatment is presented thereafter, together with some alternative options for injecting randomness.

**Russian Roulette**

The simplest and most direct way of thinking about randomness in the sand-hill crane example is to flip a coin every year \( n \) and for each of the \( y(n) \) birds in turn: head means that the bird survives, tail means that it dies. This only makes sense for a population that does not grow (\( 0 \leq R \leq 1 \)). In other words, we assume a zero birth rate \( b \) for now, and model \( 1 - d \) only, rather than \( 1 + b - d \). Births will be handled in the next Section.

With a fair coin, each bird has a 50-50 chance of survival, so \( R = 0.5 \). For different values of \( R \) between zero and one, we can think of Russian roulette, in which the revolver’s cylinder is spun anew before each bird is ... visited. By varying the number of chambers and bullets, one can achieve any desired probability \( p \) that a bird survives. The resulting recurrence is as follows:

\[
y(n) = \text{number of survivals when simulating Russian roulette } y(n-1) \text{ times.}
\]

Although this may not look much like a recurrence, it is, because \( y(n) \) is a (stochastic) function of \( y(n-1) \).

\(^{19}\)The fact that a similar result would be obtained for ratios \( y(n)/y(n-1) \) for other values of \( n \) is a bit of a coincidence, and corresponds to the fact that the growth coefficient \( R \) is assumed to be the same regardless of population size or time (that is, it is independent of both \( y(n) \) and \( n \)).
In Matlab, we can generate what is called a pseudo-random number between zero and one with the instruction

```
rand
```

Pseudo-random means that the sequence of numbers produced by repeated calls to `rand` is deterministic but hard to predict. If you restart Matlab, or, more conveniently, you call

```
rand('seed', 0)
```

then you will restart the pseudo-random generator, and obtain exactly the same sequence of numbers you obtained before the restart. So randomness is only apparent.

A good random generator (Matlab’s is good) will produce sequences with good statistical properties. One of these properties for `rand` is that the numbers being generated are *uniformly distributed* between 0 and 1. This notion will be made more precise, but it roughly means that all 64-bit binary numbers between 0 and 1 are equally likely, a bit like a fair die with $2^{64}$ faces, if you can visualize this.

You can then simulate Russian roulette with a probability $p$ of survival ($0 \leq p \leq 1$) with the instruction

```
survival = rand <= p;
```

The variable `survival` is set to 1 if the outcome of `rand` is at most $p$, and zero otherwise. Because the numbers generated by `rand` are uniformly distributed between 0 and 1, the comparison succeeds in approximately a fraction $p$ of the times, if the experiment is attempted sufficiently often. Thus, `survival` contains what is called a *random binary outcome* (“binary” because two alternatives are possible, either 0 or 1). If you need an $m \times n$ array of random binary outcomes (such as a column vector, in which case $n$ is 1), you would type

```
survival = rand(m, n) <= p;
```

Here, then, is the Matlab code for one iteration of Russian roulette:

```
y = sum(rand(y, 1) <= p);
```

The `sum` counts the number of ones (i.e., survivals) in the binary vector of length $n$ that results from the comparison `rand(y, 1) <= p`. Please make sure you understand this line of code, perhaps by typing out pieces of it in Matlab.

A full implementation of the recurrence might look like this:

```
function y = roulette(y, p)

    if p < 0 || p > 1
        error('p must be at least 0 and less than 1')
    end

    while y(end) > 0
        y(end + 1) = sum(rand(y(end), 1) <= p);
    end

(What would happen with the call `roulette(10, 1)` if the error check were not present?)

Here are a few sample runs:
An interesting question about the random binary outcomes is the following: Every time we execute the instruction

\[ y = \text{sum} \left( \text{rand}(m, 1) <= p \right); \]

with, say, \( m = 30 \) and \( p = 0.6 \) we obtain a different number \( y \). All such numbers are between 0 and 30, but are all outcomes equally likely? One way is to try the experiment, say, 10,000 times, and then tally the frequency of each of the 31 possible outcomes, that is, the number of times each number came up, divided by 10,000. The resulting plot is shown in Figure 14.

![Figure 14](image-url)

Figure 14: Frequency of occurrence of each outcome between 0 and 30 in 10,000 trials of Russian roulette with \( p = 0.6 \) and \( m = 30 \). The values of this plot add up to 1.

As expected, the outcomes are more frequently than not greater than 15, because the probability of survival is 0.6, so we can expect more than half of the 30 birds to survive. The peak of the
plot would move to the left for smaller values of $p$. The plot in Figure 14 is called a frequency distribution of the outcomes. The bell-shaped distribution in this Figure is approximately what is called a binomial distribution with parameters $m = 30$ and $p = 0.6$. The distribution would almost certainly be exactly binomial if one could run infinitely many trials, rather than just 10,000.

Because the binomial distribution is of general usefulness, it is useful to encapsulate the code that produces binomial values in a function:

```matlab
function y = binomial(m, p, n)
    if p < 0 || p > 1
        error('p must be between 0 and 1')
    end

    if nargin < 3 || isempty(n)
        n = 1;
    end

    y = sum(rand(m, n) <= p);
end
```

Here is the Matlab code that produced Figure 14:

```matlab
m = 30;
p = 0.6;
trials = 10000;

y = binomial(m, p, trials);
h = hist(y, 0:m)/trials;

values = 0:m;
clf
plot(values, h, '.','MarkerSize', 14)
hold on
for k = values + 1
    plot(values(k) * [1 1], [0 h(k)])
end
xlabel('Outcome')
ylabel('Frequency')
title(sprintf('%d trials with p = %g, m = %d', trials, p, m))
```

The plotting commands are a bit complicated, and the for loop merely draws the vertical lines in the plot. The core of this code are the two lines that compute $y$ and $h$. Each of the columns in the binary matrix `rand(m, trials) <= p` represents one trial set of 30 live/die outcomes, and the `sum` adds up the ones in each column, thereby computing the number of survivals in each of the 10,000 trials. The resulting vector $y$ has 10,000 entries (all between
0 and 30), and \( \text{hist}(y, 0:m) \) tallies into a 31-dimensional vector the number of times that \( y \) contains a value of 0, 1, ..., 30 respectively. The sum of all the tallies would be 10,000, so division by \( \text{trials} \) turns counts into fractions, that is, into frequencies of occurrence.

**Births**

Of course, random numbers can also be used to simulate births. The literature on sand-hill cranes\(^{20}\) reports that the average annual production for any adult is \( p = 0.35 \) young per year. So we need to come up with a random number generator that produces on the average \( py(n) \) positive outcomes if \( y(n) \) birds are currently alive.\(^{21}\) We could just use the Russian roulette mechanism with \( p = 0.35 \), but with a different interpretation: survival is replaced by reproduction, and the random number is *added* to the population.

However, this choice is not very satisfactory from a conceptual standpoint: using a binomial distribution with parameters \( y(n) \) and \( p \) would imply that it is *impossible* for \( y(n) \) birds to give birth to more than \( y(n) \) young in any one year. In practice, because of the small value of \( p \), the probability of this happening is very small, so the conceptual difficulty is merely theoretical. However, for more prolific species this would pose difficulties. For instance, the hare produces typically more than two litters per pair, so a bound of \( y(n) \) on the number of births in year \( n \) would be unacceptable.

To address this difficulty, we use a different distribution for births, called a *Poisson distribution*. This is closely related to the binomial distribution through a limiting process illustrated in Figure 15.

A vector of \( N \) integers that are distributed according to a Poisson distribution can be generated in various ways. For our purposes, the limiting argument above suggests a simple approximation: given a value \( \lambda = py(n) \), generate a binomial distribution with parameters \( m = \lceil M\lambda \rceil \) and \( p' = \lambda / M \) for some multiplier factor \( M \) (say, \( M = 3 \)). This will add enough of a right “tail” to the binomial to approximate a Poisson distribution for any practical purpose:

```matlab
% Returns n samples out of an approximately Poisson distribution with % parameter lambda
function y = poisson(lambda, n)
    if nargin < 2 || isempty(n)
        n = 1;
    end
    M = 3;
    m = ceil(lambda * M);
    p = lambda / m;
    y = binomial(m, p, n);
```

\(^{20}\)http://bna.birds.cornell.edu/BNA/account/Sandhill_Crane/DEMOGRAPHY_AND_POPULATIONS.html

\(^{21}\)For simplicity, we ignore the fact that birds up to three years of age do not reproduce.
Figure 15: First steps from a binomial distribution to a Poisson distribution. When doubling the number $m$ of individuals from 30 (a) to 60 (b) while keeping the parameter $p$ constant, the peak of the binomial distribution doubles from $30p = 18$ to $60p = 36$. To keep the peak in the same place (18), halve the parameter $p$ from 0.6 (b) to 0.3 (c). If this is done indefinitely, the binomial distribution (a) tends to a Poisson distribution (d) with parameter $\lambda = 30 \times 0.6 = 60 \times 0.3 = \ldots = 18$. This distribution is well defined (and nonzero) for every nonnegative integer. Of course, it becomes very small for large values of the outcome, and only a finite number of values can be plotted.
We are now ready to rewrite the recurrence for the sand-hill crane in order to account for both births and deaths:

\[ y(n) = \text{number of survivals out of a binomial distribution with } m = y(n-1) \text{ and } p = 1 - d \]

\[ + \text{ number of births out of a Poisson distribution with } \lambda = by(n-1). \]

In Matlab:

```matlab
function y = crane(birth, death, y, N)
    if birth < 0 || birth > 1 || death < 0 || death > 1
        error('birth and death rates must be between 0 and 1')
    end
    for n = 2:N
        if y(end) == 0
            break;
        end
        y(end + 1) = binomial(y(end), 1 - death) + poisson(birth * y(end));
    end
end
```

The results of several trials with different rates is shown in Figure 16.

![Figure 16](image.png)

Figure 16: Each plot shows twenty trials of a stochastic simulation of the crane population with two different combinations of birth and death rate values.

The general question we can ask of the plots thus produced is, What is the distribution of the values of \( y(n) \) at each time \( n \)? A restricted question, from which all of this Section started, would be, What is the average value of \( y(n) \) at each time \( n \)? The next obvious question seeks more detail: what is the spread of the values of \( y(n) \) at each time \( n \)? We first peek at the answers in Figure 17. The next Section introduces some of the theory on which these questions and their answers are based.
Figure 17: Means (curves) and standard deviations (bars) of the values in the two sets of plots in Figure 16.