Chapter 1: Sampling, Resampling, and Inference

If one could go out and measure exactly and completely the quantities one is interested in, there would be no need for statistics. There are in fact many cases where this can be done; perhaps this is why so many people get so far in life without knowing anything about statistics. Want to know whether you have a fever? Use a thermometer to take your temperature. Want to know the price of an item in a store? Look at the price tag or ask the clerk. Want to know how many people there are in the United States? Go out and count them, a procedure stipulated by the census provisions of the U.S. constitution to be performed every 10 years.

In reality, the census situation is not so simple. A count of approximately 280 million U.S. residents cannot be performed instantly. In the process of doing it some people who have already been counted will die, and others will be born without being counted. Some people will be counted twice since they will move from one residence to another during the counting period; others will not be counted for the same reasons. People in some segments of the population — homeless people, illegal immigrants — are just hard to count. For these reasons, the census bureau has recently proposed, in the face of considerable controversy and opposition, not to make an explicit person-by-person count, but to sample the population and use statistical techniques to estimate the size of the population.

In many cases we have no choice but to base our conclusions on a *sample* rather than a full measurement. Suppose we want to find out which of two treatments for breast cancer is more effective. It would be impossible to apply both treatments to every person who has breast cancer. Instead, we select a sample of people with the disease and apply one treatment. We take another sample — different people — and apply the other treatment. We then compare the treatment outcomes in the two sample groups.

This procedure raises some important practical questions. How should the two samples be picked? How large should the samples be? If we do find a difference between the treatment outcomes in the two groups, how confident are we that it is not just a chance outcome, the way a flip of the coin will randomly favor one player or the other? If we find no difference in outcome, how sure are we that this is not because our sample groups are too small?

These are all questions of *statistical inference*: how we reason from a sample to the entire population of interest. Some other examples:

- A biologist studies the ecology of freshwater mussels. He is interested in whether the diversity of species is decreasing in the face of deterioration of the environment from pollution and from the introduction of rapidly proliferating non-native species such as the zebra mussel. He cannot conduct a complete census: doing so would kill all the mussels. Instead, he takes small samples and infers from the sample what is happening to the population as a whole. For example: if 8 different species are found this year as opposed to 12 in the previous year, is one justified in concluding that species diversity is falling?

- The price of health insurance is based on a calculation of how likely you are to get sick and how much it will cost to treat you if you do fall ill. The estimate of sickness rates is not primarily based on your own personal history (although factors such as smoking, age, gender, and so on play a role) but on data collected from a sample of the population. Insurance companies and health maintenance organizations need to use data from a past sample of the population in order to make conclusions about their present and future customers. How much money does the insurance company need to have in reserve in order to ensure that it can pay the bills for its customers?

1.1 Sampling

We sample from a population[1] in order to make inferences about the population as a whole. How are we to collect the sample? What are the consequences of sampling for the inferences?

People are often surprised to hear that the best way to pick samples is randomly. The process has two steps. (There are more complicated arrangements, but we won't consider them here.)

Step 1: Identify and delineate the population of interest.

This identification is not done at random and often involves some expert knowledge about the subject of study. In a political poll, for instance, the population might consist of people who answer their phones and who respond affirmatively when asked if they are a registered voter. In a study of cancer treatments, the population might consist of all the patients who present themselves at one of the participating clinics and who are diagnosed as having a particular type of tumor.

Step 2: Pick the sample at random from the population identified in Step 1.

Care must be taken to ensure that the sample really is taken at random. In a cancer experiment, the population might be the first 1000 qualified patients who happen to come to the clinic and who sign a form indicating willingness to participate in the experiment.[2] Then, for each such patient, a computer is used to generate a random code indicating whether the patient will be in the treatment group or in the control group.

Television and radio stations sometimes conduct polls by asking their audience to telephone the station. Such polls say little about the overall population; the respondants are self-selected rather than randomly selected and the self-selected respondants tend to be those with strong opinions. The result can be a poll that is wildly misleading about the overall population.

[1] The word "population" suggests that we are talking only about people or animals. In statistics, though, "population" has a broader meaning and can refer to any collection.

[2] This constitutes a random sampling from a particular population: those people whose go for treatment to cancer clinics. If, however, the clinic announces its planned experiment publically, and patients who have previously proved untreatable converge on the clinic, the sampling would not be randomly drawn from all patients seeking cancer treatment.

Random sampling is important because it helps in avoiding the systematic influence of confounding variables. A famous example is the 1954 polio study as described in reference [?]. As part of this study, the parents of children in second grade were asked to give permission for their children to be injected with an experimental polio vaccine. The children whose parents said yes were given the vaccine. The children whose parents said no were put into the control group. This is not random sampling; the two groups differ systematically in terms of the confounding variable of parental permission.

You might find it hard to believe that parental permission has anything to do with polio. As things turned out, however, there was a strong connection between the willingness of parents to give permission and the risk of polio.[3] How do we know in actuality that parental permission is associated with risk of polio? Because there was a simultaneous second study which randomly selected children for actual vaccination from the sub-group of children whose parents gave permission; the other children with permission were injected with a sterile placebo.

One important consequence of using random sampling is that we encounter *sampling variability*: different samples provide somewhat different results. Consider a thought experiment in which we imagine a population of 1000 people, 10 of whom have a certain genetic trait — a 1% prevalence rate. Take a random sample of 100 people from the population. This sample might have no people with the trait, or one, or two, or up to 10. The exact number is random. Suppose that the sample has 2 people with the trait. If we knew only about the sample, and not about the whole population, would we be justified in concluding that the population has a 2% prevalence rate? To some extent the answer is yes, but we also realize that the rate in the population might not be exactly the same as the rate in our random sample. In addition to information about our sample itself we also have other knowledge. This is our knowledge of the sampling process itself and the way it leads to random variability.

One of the main tasks of statistical inference is to characterize sampling variability in a way that puts reasonable bounds on what we can conclude about the population from our sample of it. For instance, if our sample of 100 shows a 2% prevalence, we would be quite confident that

[3] This has been explained as resulting from the higher educational levels of parents who give permission. (See reference [?].) This is associated with more sanitary living conditions and consequently a reduced exposure to the polio virus during early childhood. An early exposure to polio can result in a natural immunity without noticible symptoms.

the prevalence in the population as a whole is not 90% but we might not be so sure that is isn't 5%.

By using random sampling when collecting data we gain the ability to use mathematical techniques to analyze the sampling variability. Often, we simply assume that the sample is random because this makes it easy to do statistical inference calculations. Of course, to the extent that the assumption is wrong the results of our calculations may be misleading and unreliable.

Our use of random sampling produces an uncertainty about our results that can be quantified and analyzed objectively. Failure to use random sampling results in uncertainties that are themselves unknown and subjective, and leads overall to a higher level of ignorance.

1.2 Resampling

The basic idea of resampling is very simple: mimic the process of sampling by picking another sample at random from a hypothetical population of interest, usually based on data from your sample. Much of what we study in statistical inference is variability introduced by taking a sample of a population. When it is too expensive or impractical to sample more data from the population itself, we can study sampling variability by a simple expedient: we sample instead from an artificial population constructed on our computer and that embodies everything we know about the population of interest. In many, but not all, of the examples that follow, this artificial population is the very data set from which we seek to draw inferences. Since the data set is itself a sample of the whole population, we are taking a sample from the sample: resampling. This doesn't, of course, provide more information about the population, but it does provide us with a way of understanding the consequences of sampling variability for drawing inferences about the population based on our data.

One problem in mimicking the sampling process is that generally the population is much bigger than the sample. When carrying out resampling, though, our data is often exactly the same size of the resample to be taken. In order to make the data set seem larger than it is, we resample *with replacement.* For example, when resampling 10 points from a data set of 10 values, we keep all 10 values available when selecting each of the resampled points. It is as if we wrote all 10 values on individual slips of paper and put them in a hat. To resample 10 points, we select one slip at random from the hat and write its value on a tally sheet. We than replace the slip in the hat and repeat the process until we have

10 values on the tally sheet. This means that some of the original data values might appear more than once on the tally sheet; others might appear not at all.

Based on this simple process of resampling, a number of techniques have been developed for answering questions in statistical inference. It is only in the last 20 years that resampling has achieved widespread use. This is actually a substantial fraction of the 90 or so years since the advent of modern statistical inference, but historically it is true that most statisticians have been trained and most statistics students have learned non-resampling ways of doing statistical inference. Resampling is generally impractical without access to computers. But fast, inexpensive computing is now a fact of life and this allows us to exploit an important advantage of resampling in teaching statistics: it is a natural and intuitive way of solving problems in statistical inference.

1.3 An Introduction to Probability

We are surrounded in everyday life by statements involving probability. Most people have learned how to translate these statements into operational terms. "The chance of rain is 10% today" means that it is OK to plan a picnic. "This surgery has a survival rate of 98%" means that the surgery is risky, but that it is worthwhile if it will resolve a fatal disorder. "The probability of a fatal car accident is $\frac{1}{5{,}000{,}000}$ per vehicle mile" means that driving a car is fairly safe. "The odds of winning in the PowerBall lottery are 1 in 180 million" means not to quit your job because you think your ticket is lucky.

Where people have difficulty is in using probabalities to make quantitative judgements or to analyze complicated situations. This is where Resampling Stats can help.

Where do probabilities come from?

When flipping a coin, what is the probability of getting heads? Everyone knows that heads and tails are equally likely: the probability of each is 50%. How do we know this? We could test the hypothesis by flipping the coin many times (10? 100? 1000?) and seeing whether heads comes up about half the time. But few of us have ever done this and certainly we are not in the habit of vetting the coin to be used in a game by testing it many times. We know that the probability of heads is 50% because 1) this seems reasonable ("There's no difference between heads and tails from the coin's perspective, so why should one or the other come up more often?") and 2) we have never read anywhere an exposé of the

50% hypothesis.

Similarly, the probability of getting a 2 in a roll of a single die, or of drawing the ace of spades from a normal 52 card deck are 1/6 and 1/52 respectively. These probabilities are established by tradition, and have not been contradicted by personal experience. They are aided by our informed intuition that if there are N possible outcomes, each of which is deemed to be equally likely, then the probability of any single outcome is $1/N$: with cards $N = 52$; with a die $N = 6$; with a coin $N = 2$.

Probabilities are quite commonly assigned in a different way: by performing many trials and counting events. If 9342 coronary bypass operations are performed and the patient dies in 274 of them, the probability of dying during the operation is taken to be $\frac{274}{9342}$.

Sometimes it is impossible to conduct trials and count events. How do we decide what is the probability of rain today in our home town? It would not be easy to construct 100 duplicate Earths — all having the same conditions — and see how many of them have rain in our town. Instead, weather forecasters look back through the historical records to see in what fraction of similar days there has been rain. What does "similar" mean? The same season; same temperature, humidity and air pressure; same pattern of warm and cold fronts. Or, forecasters can take several different computer models and see in what fraction of them rain occurs.

Often, probabilities are just made up to reflect our best guess. What's the probability that we will be hired for a desirable job? "I'm one of two people interviewed, and I think I have a pretty strong record, so the probablity is 1/2 or maybe even 2/3." Such subjective assessments can be useful. Even without knowing the detailed qualifications of competing candidates or the internal politics of a company, it is reasonable to say that our chances of getting the job are much better if we are one of two candidates than if we are one of 25 candidates.

Finally, for our purposes in using Resampling Stats it is important to know about computers and probabilities. We will be generating random numbers on the computer and making certain assumptions about probabilities. For example, if we sample one number from a list of 35 numbers using the computer's random number generator to make the selection, how do we know that each of the numbers is equally likely to be selected. One answer is that we can test this hypothesis by performing many trials and seeing whether each of the numbers is selected about the same number of times. This is quite easy to do on the computer and is the topic of Example **??**. Another answer is that the computer random number generator has been carefully designed and extensively tested.

Example 1: Rolling a Die

Let's conduct a test of the computer random-number generator using the Resampling Stats software. Start up the MATLAB software and, if you have not already done so, install the Resampling Stats software as described in Appendix ??. You may also want to work through the first few sections of the tutorial introduction to MATLAB found in Appendix ??.

When you start the MATLAB program, you will be prompted by a ≫ to enter your commands. (You don't type the ≫, that's the computer's job.) After you type your command, press the ENTER key and MATLAB will process[4] your command and print the results. If you end the command with a semi-colon, "`;`", MATLAB won't print the result of the command but will execute it just the same. For example,

```
≫  3+4      Press Enter
  ans:     7
```

To test the random number generator, we will simulate the roll of a die. Or, rather, we will simulate 6000 rolls of a die.

Our first command will create a variable `die` that models the outcomes of a die toss:

```
≫  die = [1 2 3 4 5 6]
  die:     1 2 3 4 5 6
```

This is an extremely simple command that just creates the variable `die` and assigns it a value. The value is the list of numbers one through six.

Next, we will simulate 6000 rolls of the die and put the results in the variable `rolls`. We use the Resampling Stats function SAMPLE[5] to draw the 6000 samples from `die`:

```
≫  rolls = sample(6000, die)
  rolls:     4 3 4 3 3 2 4 5 4 4 2 3
             5 5 3 4 5 1 4 1 3 2 6 1
             5 6 6 5 3 3 2 4 and so on
```

The first roll was a 4, the second a 3, the third a 4, and so on. If you typed this example yourself, you probably got different numbers. This is because the program SAMPLE uses the computer random number generator to draw each number from a list like `die` at random and with equal probability. (Later, we'll see other ways that SAMPLE can be used.)

It might take a few seconds for all the numbers to print to your screen.

[4] Synonyms for "process" are "evaluate" and "execute."

[5] Throughout this book, a program name written in the SMALL CAPITALS FONT means that documentation for the program is in the reference section of the book.

Since the results of the sample are contained in the variable `rolls`, it's more convenient to suppress the printing with a semi-colon:

```
≫  rolls = sample(6000, die);
```

There's nothing special about the name `rolls`. For example, we could have named the results `data` since these 6000 numbers are the data which we will use to test the computer's random number generator.

There are many ways that we might want to test the numbers in `rolls` for randomness. The most obvious is to count how many times each of the possible outcomes, 1 through 6, appears. Since each outcome is equally likely, we expect to find each number occurring $\frac{1}{6}$ of the time. Out of 6000 rolls, each number should appear 1000 times.

The Resampling Stats function MULTIPLES counts how many times each outcome appears:

```
≫  results = multiples(rolls)
  results:     1014 958 986 995 1055 992
```

(MATLAB prints the output as a column of numbers, but we have written it as a row in order to save paper.) MULTIPLES tells us that a 1 appeared 1014 times, a 2 appeared 958 times, and so on. Again, if you typed this line yourself, you probably got somewhat different numbers because the data were generated randomly. We expected to see each outcome about 1/6 of the time, or in this case exactly 1000 times. This isn't exactly what happened, since the sample is being drawn at random. But the result are quite close to 1000 in each case. Deciding whether the results are close enough to 1/6 is itself a more advanced problem in statistical inference, and is the subject of Example **??**. But for now it looks like the simulated die behaves more or less as expected.

You should be aware that computer random number generators are not really random. If you know enough about the random number generator, you can predict perfectly what numbers it will generate. In using the computer, we adopt a very subjective point of view: we do not know the details of the random number generator and so to us it appears random. If you run the example above again, you will (probably) get somewhat different results. Although there are some quite obscure cases where this subjective point of view fails us (particularly when the random number generator has been improperly designed), for our purposes the assumption of randomness is quite safe. (But see the documentation for the random number generator SEED.)

Computing Probabilities

In many games we roll two dice and add together their individual outcomes. There are 11 possible outcomes: the numbers 2 through 12. Are these 11 outcomes equally likely? Informed dice players know that the answer is "No." Let's simulate the situation. Let's create another die

```
≫  die2 = [1 2 3 4 5 6];
```

and make a roll of each — that is, take one sample from each — adding up the results

```
≫  roll = sample(1,die) + sample(1,die2)
```

That's one sample roll. We want to make many rolls and keep track of the results. In Resampling Stats we can keep track of results with the STARTTALLY and TALLY functions, and we repeat the sample many times using the MATLAB `for` command. The file `twodice.m` contains a sequence of commands. (See section **??**.) By typing the command

```
≫  twodice
```

MATLAB evaluates all of the commands in the file.

```
1   results = starttally;
2   for trials = 1:1000
3      roll = sample(1, die) + sample(1, die2);
4      tally roll results;
5   end
```

We have numbered the lines here to make it easier to explain what each line does. The line numbers don't appear in the actual commands.

Line 1 This statement tells MATLAB that we want to start keeping tally of the results. We'll give the name `results` to the variable that holds the results.

Line 2 Do 1000 trials. All the commands up to the `end` will be run 1000 times.

Line 3 Add together the random sample from each of the two dice.

Line 4 Puts the sum from the current trial into the list of results.

Line 5 Go on to the next trial.

After executing the commands in the file `twodice.m` with the single command

```
≫  twodice
```

the variable `results` will contain the results of the 1000 simulated rolls of the dice. Note that while the command is being executed the cursor changes to an hourglass form.

We can tabulate the results:

```
≫   [howmany, multiples] = multiples(results)
```

Here, we're asking `multiples` to give us information about what were the values and how often they occurred (`howmany`). Having both these things makes it easy to print out the results in the form of a table. We divide `howmany` by the number of trials (1000 in this case) to convert to a probability.

```
≫   [values, howmany/1000]
  ans:     2.0 0.0240
           3.0 0.0490
           4.0 0.0730
           5.0 0.1290
           6.0 0.1410
           7.0 0.1680
           8.0 0.1320
           9.0 0.1060
           10.0 0.1000
           11.0 0.0560
           12.0 0.0220
```

We can see that 7 is the most likely outcome and that 2 and 12 are quite unlikely.

Another simple example: what is the probability of getting 40 or fewer heads when flipping 100 coins? In the file `coinflip.m` we see that the method of solving this problem using Resampling Stats is very similar to that used for the two dice:

```
% the text following the % sign is a comment.
% probability of 40 or fewer heads on 100 coin flips.
% we will represent the coin as 0 for heads, 1 for tails
coin = [0 1];
z = starttally;
for trials = 1:1000  % do 1000 trials
  % the experiment is to flip 100 coins and
  % count how many 'heads' (coded as 0)
  flips = sample(100, coin);
  nheads = count(flips == 0);
  % Note: == means 'test for equality'
  %        = means 'assign value to variable'

  % Now tally the results for this trial
  tally nheads z;
end
```

Now the variable `z` contains the number of heads in each trial of 100 flips. We'll find out what proportion of these trials had 40 or fewer heads:

```
≫   proportion( z <= 40 )
  ans:    0.024
```

If you have studied probability then you know that the problem of the dice can easily be solved using counting techniques. (For example, there are 6 different ways of getting a sum of 7, the probablity of each of the ways is 1/36, so the probability of getting a 7 is 6/36 = 1/6.) The coin flip problem can be solved using the binomial distribution. Using a simulation, we have closely matched these probabilities, but because of sampling variability the match is not exact. (We can make the match better by using more trials. See Sec. **??**.) Why use the simulation if it provides only an approximation to the true answer? One reason is that we use the same kind of simulation in both cases, rather than having to learn and remember specialized information. Of course, this is not an advantage if we already know the specialized information, but one of the objectives of Resampling Stats is to give novices the ability to solve probability problems. Even for people who are expert in solving problems in probability, simulations provide a way of checking the answers they got in other ways; conducting a simulation is for most people easier and less error prone than than solving the problem using algebraic techniques.

Another, more fundamental reason for using simulation is that for many kinds of problems the answer cannot be found in statistics textbooks; using simulation is the only practical way to solve the problem.

Example 2: The Birthday Problem

What is the probability that two or more people in a group of 30 have the same birthday? This is a famous problem, and is actually quite difficult using conventional methods. The probability is surprisingly large, and is easily found using resampling. The complete program is in `birthday.m`, but here we work through a single trial.

Step 1: Take a sample of 30 birthdays randomly from the 365 days of the year. [6]

[6] `1:365` means "1 to 365." See the tutorial in Sec. **??**, Step **??**.

```
≫   days = sample(30, 1:365);
  days:     25 208 198 166 332 205 170 20 62 234 240 273
            139 87 170 175 90 263 242 243 283 81 345 303
            351 211 121 159 90 158
```

Step 2: Count how many times each day appears in the sample

```
≫   b = multiples(days)
  b:     1 1 1 1 1 2 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1
         1 1 1
```

In this trial, there were two different pairs of people who had the same birthday.

Step 3: Compute whether there were any shared birthdays. Of course, it's easy to see this by looking at the result from Step 2, but we need to tell the computer how to do this.

```
≫   result = any(b>=2)
  result:     1
```

The answer `1` (standing for "true") indicates that there was indeed a shared birthday. If there had been no shared birthdays, the answer would have been `0` (standing for "false").

We found, in this single trial, that there were two people with the same birthday in this group of 30 — in fact there were 2 shared birthdays. On another trial, the result might be different. By conducting many trials we can estimate the probability of two or more matching birthdays in the group.

But before we do this, let's generalize the problem a little bit. Let's allow you to specify how big the group is, and not restrict the problem to exactly 30. In addition, we'll allow you to specify how big should be the group of people sharing a birthday — this was 2 in the above case. In order to generalize the commands, we write the commands in an m-file function, as described in Step **??** of the MATLAB tutorial in Appendix **??**.

The m-file `birthday.m` looks like this:

```
function res = birthday( groupsize, cliquesize )
% BIRTHDAY --- gives the probability of more than one person in a
% group having the same birthday.
% birthday( groupsize, cliquesize )
% groupsize  -- size of the group
% cliquesize -- how many people have the same birthday
% example: birthday(30,2)
% gives the probability that 2 or more people
% have the same birthday in a group of 30

z = starttally;
for trials = 1:1000
  d = sample(groupsize, 1:365 );
  b = multiples(d);
  f = any(  b > cliquesize );
  tally f z;
end

res = proportion(z == 1);
```

The first line of this file tell MATLAB to create a function that takes two arguments, `groupsize` and `cliquesize`. The following lines are comments that will be printed if you give the command

```
≫  help birthday
```

The rest of the file is the MATLAB commands that carry out the repeated trials. The last line says that the returned value from the function is `proportion(z==1)`. The value returned by the function is the value held by the variable, in this case `res`, whose name appears after the keyword `function` in the first line of the file. (See Section **??** Step **??**.)

We can use BIRTHDAY to find the probability of 2 or more people in a group of 30 share a birthday.

```
≫  birthday(30,2)
   ans:     0.714
```

Is the probability larger than you would have expected? Since BIRTHDAY has been written as a function, you can use it, for example, to find the smallest group that has more than a 50% probability of having 2 or more people with a shared birthday. Or, to find out the probability that there will be a clique of at least 3 people having the same birthday out of a group of 100, give the command

```
≫  birthday(100,3)
   ans:     0.63
```

Random Variables

For some probability problems, the outcome is one of a set of non-numerical possibilities: heads or tails for a coin flip; rain, sun, cloudiness, or snow for a weather prediction. For other probability problems, the outcome is a number: the amount of money you win or lose after each spin in roulette; the temperature tomorrow; the time until failure of a disk drive in a computer; the number of people trying to use the telephone network at any one time. Such numerically valued random numbers are called *random variables*.

Example 3: Annual Rainfall

As an example of a random variable, consider the problem of finding the total amount of rain (in inches) each year in a location where the probabilities of the amount of rain each day are known. Let's imagine a place where there is no rain on 80% of days, .25 inches on 10% of days, .5 inch on 5% of days, and 1 inch on 5% of days. Notice that there are 4 different possible events: no rain; .25 inches of rain; .5 inches; 1 inch. These events are not equally likely, so we need to specify to the computer the probability of each event. We convey the probability information to MATLAB in the form of a matrix:

```
≫  dailyrain = [ 80 0.0; 10 0.25; 5 0.5; 5 1.0];
```

The matrix has a specific form: for each event, give the probability of the event followed by the numerical value for that event. For example, "no rain" has a numerical value of 0.0. The different events are separated by semi-colons (;). If we print out `dailyrain`, the matrix structure is more evident

```
≫  dailyrain
  ans:     80 0.00
           10 0.25
           5 0.50
           5 1.00
```

If we assume that the amount of rain each day is independent of the amount of rain on any other day[7] we can easily find the total amount of rain in one year, by generating samples for 365 days and adding them up:

```
≫  annualrain = sum( sample( 365, dailyrain ) );
  ans:     43
```

But this is just a sample for one year of the rainfall. Just as the amount of rain each day is a random variable, so is the amount of rain each year. We can estimate the probability of any given amount of rain in a year

[7] This is an unrealistic assumption that could be avoided by collecting more information and specifying a more complex structure to the problem, but we'll keep this example simple.

by repeating the simulation many times and making a histogram of the results, as in the file `rainyear.m`

```
dailyrain = [80 0.0; 10 0.25; 5 0.5; 5 1.0];
annualrain = starttally;  % save the results in annualrain
for trials=1:1000
  oneyear = sum( sample(365, dailyrain) );
  tally oneyear annualrain;
end
```

After running this script, we can look at the samples of annual rainfall contained in the tallying variable `annualrain`

```
≫   annualrain
   ans:     33.75 33.50 29.00 42.50 28.00 29.75 37.75
            40.50 28.75 29.75 35.50 37.25 39.75 42.75
            31.25 41.75 47.50 and so on
```

We see that the values are typically around 40 inches per year.

Describing Random Variables

Describing a random variable by printing a long list of samples is not very concise, and it is hard for people to gather much insight from a long list of numbers.[8] Making a histogram of the data gives a quick visual impression of the probability distribution of the random variable. Fig. **??** shows a histogram of the simulated annual rainfall data. This histogram was made with the command

```
≫   histogram(annualrain, 'Annual Rainfall')
```

From the histogram, we can easily see that the rainfal is typically around 35-40 inches/year, and sometimes falls as low as 20-25 inches/year or rises to 50-55 inches/year.

Figure 1:
A histogram of the amount of rain in a year
in a hypothetical location.

Although the histogram conveys a lot of information, it is convenient to summarize a random variable in more compact terms. Even the most compact and rudimentary description of a random variable requires two numbers: one number to describe the center of the distribution and another to describe the width of the distribution.

[8] On the other hand, from the computer's perspective a long list of samples is an excellent way to describe a distribution. This is, in fact, the idea behind resampling.

The most common ways of describing the center of the distribution of a random variable are the *mean* and the *median.* As is well known, the mean is the average outcome of the random variable. The median is the value which is smaller than 50% of the outcomes and larger than the other 50%. Try

```
≫  mean(annualrain) ⇒ ans:  36.60
≫  median(annualrain) ⇒ ans:  36.75
```

In this case, the median and the mean are almost the same, but that is not always true.

There are several widely-used ways to describe numerically the width of a random variable's distribution:

Standard Deviation The most common way of describing the width. It tells how far, on average, each outcome is from the mean.

```
≫  std(annualrain) ⇒ ans:  4.77
```

Total Range The difference between the maximum value in the data and the minimum value

```
≫  max(annualrain) - min(annualrain) ⇒ ans:  33
```

Interquartile range The difference between the 25% and 75% percentile

```
≫  percentile(annualrain,.75)-percentile(annualrain,.25)
```

This calculation is easy enough, but to make it simpler there is a function `iqr` that saves some typing:

```
≫  iqr(annualrain) ⇒ ans:  6.5
```

The interquartile range is often preferred to the total range, since the most extreme values in a data set are sometimes outliers that do not represent the data as a whole.

Percentile ranges Often it makes sense to consider the 5% and 95% percentiles or, most commonly, the 2.5% and 97.5% percentiles to give a range that covers 95% of the data

```
≫  percentile(annualrain,[.975 .025])
  ans:    27.250 45.875
```

For probability problems that don't involve random variables (remember, a random variable is a random event that has a numerical outcome), it doesn't make sense to talk about means, medians, standard deviations, and so on. For example, what is the mean value of the two possible outcomes, heads or tails, of a coin flip? Hails? Taads?