Chapter 2
Randomness and Probability

Suppose that, for some reason, we want to know how many times a second-language learner makes errors in a writing task; to be more specific, let's assume we will only count verb inflection errors. The dependent variable (here, the number of inflection errors) is random in the sense that we don't know in advance exactly what its value will be each time we assign a writing task to our subject. The starting point for us is the question: What's the pattern of variability (assuming there is any) in the dependent variable?

The key idea for inferential statistics is as follows: If we know what a 'random' distribution looks like, we can tell random variation from non-random variation.

As we will see, many random phenomena have the following property: while they are unpredictable in specific individual cases, they follow predictable laws in the aggregate. Once we learn to identify this 'pattern of chance,' we can confidently distinguish it from patterns that are not due to random phenomena.

In this chapter and the next, we are going to pursue this key idea in detail. Our goal here is to look at distribution patterns in random variation (and to learn some R on the side). Before we get to this goal, we need know a little bit about probability theory, so let's look at that first.

2.1 Elementary Probability Theory

2.1.1 The Sum and Product Rules

We will first go over two very basic facts from probability theory: the sum and product rules. Amazingly, these are the only two facts we need for the entire book. The basic idea is that, just as in propositional logic we build up complex propositions from basic ones and compute the truth value of

the whole from the values of the parts, so in probability theory we build up complex *events* from simple ones and compute the probability of the whole event from the probabilities of the parts, using the sum and product rules. Keep this compositional property in mind as you work through this chapter.

We are going to present these ideas completely informally. There are very good books that cover more detail; in particular we would recommend *Introduction to Probability* by Charles M. Grinstead and J. Laurie Snell. The book is available online from the website

http://www.dartmouth.edu/~chance/.

For the present book you do not need to know anything more than what we discuss here. However, a time will come when you will want to read up more on probability theory; the above-mentioned book would be a useful starting point.

Consider the toss of a fair coin, which has two sides, H(eads) and T(ails). Suppose we toss the coin once. What is the probability of an H, or a T? You might say, 0.5, but why do you say that? You are positing a theoretical value based on your prior expectations or beliefs about that coin. (We leave aside the possibility that the coin lands on its side.) We will represent this prior expectation by saying that $P(H) = P(T) = \frac{1}{2}$.

Now consider what all the logically possible outcomes are: an H or a T. What's the probability that either one or the other of these will occur when we toss a coin? Of course, you'd say, 1; we're one hundred percent certain it's going to be an H or a T. We can express this intuition as an equation, as the sum of two mutually exclusive events:

$$P(H) + P(T) = 1 \qquad (2.1)$$

There are two things to note here. One is that the two events are *mutually exclusive*; you can't have an H and a T in any one coin toss. The second is that these two events exhaust all the logical possibilities in this example. The important thing to note is that **the probability of mutually exclusive events occurring is the sum of the probabilities of each of the events**. This is called the SUM RULE.

To understand this idea better, think of a fair six-sided die. The probability of each side s is $\frac{1}{6}$. If you toss the die once, what is the probability of getting an odd number? The event 'getting an odd number' can be broken down into the mutually exclusive events 'getting a 1, or a 3, or a 5' and so the answer is $\frac{1}{6} + \frac{1}{6} + \frac{1}{6} = \frac{1}{2}$.

Suppose now that we have not one but *two* fair coins and we toss each one once. What are the logical possibilities now? In other words, what sequences of heads and tails are possible? You'll agree that the answer is: HH, HT, TH, TT, and also that *all of these are equiprobable*. In other words: $P(HH) = P(HT) = P(TH) = P(TT)$. There are four possible EVENTS and each is equally likely. This implies that the probability of each of these is $P(HH) = P(HT) = P(TH) = P(TT) = \frac{1}{4}$. If you see this intuitively, you also

understand intuitively the concept of PROBABILITY MASS. As the word 'mass' suggests, we have redistributed the total 'weight' (1) equally over all the logically possible outcomes (there are 4 of them).

Now consider this: the probability of any one coin landing an H is $\frac{1}{2}$, and of landing a T is also $\frac{1}{2}$. Suppose we toss the two coins one after another as discussed above. What is the probability of getting an H with the first coin followed by a T in the second coin? We could look back to the previous paragraph and decide the answer is $\frac{1}{4}$. But probability theory has a rule that gives you a way of calculating the probability of this event:

$$P(H) \times P(T) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4} \tag{2.2}$$

In this situation, an H in the first coin and an H or T in the second are completely independent events—one event cannot influence the other's outcome. This is the PRODUCT RULE, which says that **when two or more events are independent, the probability of both of them occurring is the product of their individual probabilities**.

And that's all we need to know for this book. At this point, you may want to try solving a simple probability problem: Suppose we toss three coins; what are the probabilities of getting 0, 1, 2, and 3 heads?

2.1.2 Stones and Rain: A Variant on the Coin-toss Problem

Having mastered the two facts we need from probability theory, we now begin our study of randomness and uncertainty, using SIMULATIONS. When you repeatedly carry out random simulations, you will usually get a different answer each time. Repeatedly running such a simulation (performing many REPLICATES) gives you a good intuitive sense of how patterns change or remain stable on each run, and the extent to which they change. We encourage you to run simulations more than once, to see these effects for yourself. We also encourage you to change the values given to the functions (altering the input probabilities, changing the sample size, etc.). This is the great advantage of using a language like R to explore statistical concepts.

Because the coin example is so tired and over-used, we will use a different example of a random process (due originally to Polya, 1954, 56) for purposes of our discussion of probability. Suppose we have two identical stones (labeled L(eft) and R(ight)), and some rain falling on them. We will now create an artificial world in R and 'observe' the raindrops falling randomly on the stones. We can simulate this quite easily in R using the built-in random generator function `rbinom()`, which takes three arguments:

```
> rbinom(1, 1, 0.5)
```

```
[1] 1
```

The above command says that, in a world in which the probability of a R-stone hit is 0.5 (a reasonable assumption given the stones are the same size), generate a raindrop on one pair of stones (the second argument), and do this once (the first argument). Return how many times we successfully get such a hit. So a return value of 1 means the drop landed on the Right stone, 0 means it didn't.

We can repeat this experiment 10 times as follows:

```
> rbinom(10, 1, 0.5)

 [1] 1 1 1 1 1 1 0 1 1 1
```

Notice that different repetitions give different results (it is a random process after all):

```
> rbinom(10, 1, 0.5)

 [1] 0 0 0 1 1 0 0 1 0 0
```

Intuitively, if the probability of a success is 0.5, then this should be reflected in the proportion of successes we see in our observations:

```
> sum(rbinom(10, 1, 0.5))/10

[1] 0.3
```

And if we increase the number of replicates, this proportion gets closer and closer to the actual probability underlying the process:

```
> sum(rbinom(100, 1, 0.5))/100

[1] 0.46

> sum(rbinom(1000, 1, 0.5))/1000

[1] 0.478

> sum(rbinom(1e+06, 1, 0.5))/1e+06

[1] 0.500579
```

In our next experiment, we will observe a section of pavement consisting of 40 pairs of stones, and record the total number of successes (Right-stone hits) for that section (we just keep track of where the first drop falls in each case). Note that we could just as well observe one pair of stones 40 times: the important point is that our single experiment now consists of a group of 40 distinct and independent TRIALS.

```
> rbinom(1, 40, 0.5)
```

```
[1] 24
```

Note that there are many different OUTCOMES of the experiment that could have given rise to the hit-count we just generated. R is not telling us the pattern of Right-stone hits on the section of pavement (or the particular sequence of hits and misses on one pair of stones): it is simply reporting the count, which is usually all we are interested in. Many different patterns have the same count. (We will return to the structure of the individual outcomes later).

Intuitively, since the probability is 0.5, we would expect about half (i.e., 20) stones to register successful hits in any one experiment. But notice too that there are actually 41 *possible* counts we could see: it is possible (though we think it unlikely) that in our observation of 40 stones, we see just 3, or as many as 39, successful hits. We can get a feel for this if we repeat this experiment a number of times:

```
> rbinom(10, 40, 0.5)

 [1] 19 23 15 20 22 19 22 25 22 20
```

The number of successes moves around across a range of numbers (it is a random process after all). You may be surprised to see how seldom we observe 20 hits (in a short time we will be able to compute the exact probability of this occurring). But also notice that it tends not to move across the entire possible range (0–40). In fact, it moves across a range of numbers that are rather close together, and 'balanced' around a 'central' number. The process is random in the details, but has a predictable pattern in the mass. Let's do more random simulations to explore this further.

First we create a vector to hold the results of 1000 experiments:

```
> results <- rbinom(1000, 40, 0.5)
```

Let us look at a histogram of these results:

```
> hist(results, breaks = 0:40)
```

Figure 2.1 shows the result of the above command. The central number and the balance of the distribution become apparent: the most frequently occurring value in this list is (about) 20, and it is centered and balanced in the sense that the frequency of values above and below it fall off symmetrically.

The extent to which this pattern is apparent depends on the number of times we repeat (replicate) the experiment. Let's replicate this experiment *i* times, where *i* =15,25,100,1000. Each time we will plot the histogram of the results.

The code for plotting the distributions is shown below:

```
> multiplot(2, 2)
> p <- 0.5
```

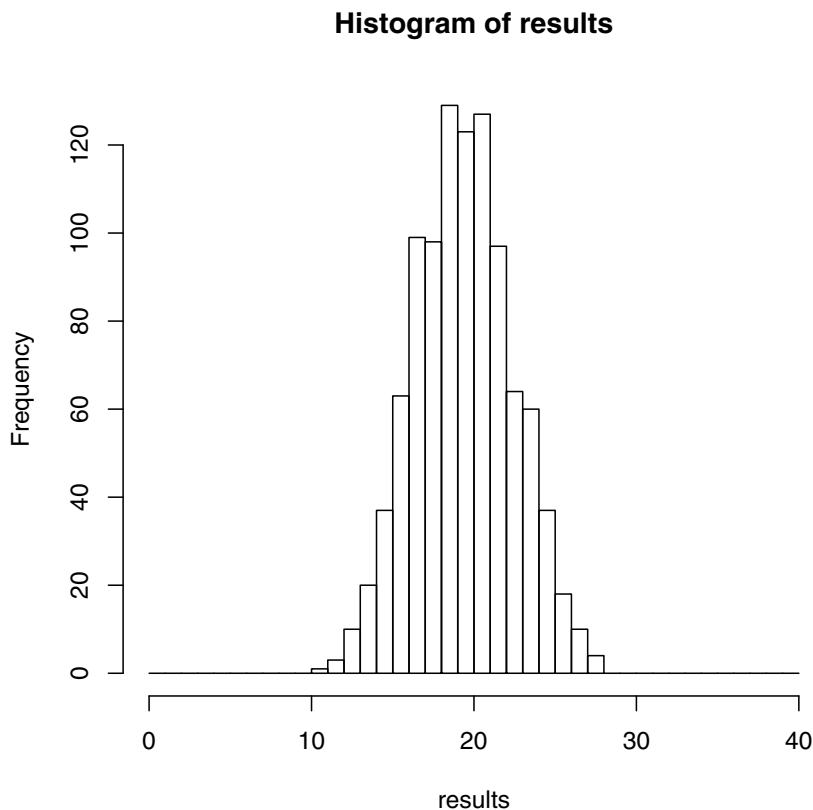**Histogram of results**

Fig. 2.1 The frequency histogram of the result of observing 40 drops 1000 times.

```
> drops <- 40
> replicates <- c(15, 25, 100, 1000)
> for (i in replicates) {
      results <- rbinom(i, drops, p)
      title <- paste(c("No. Obs.", i, sep = " "))
      hist(results, breaks = 0:40, ylab = "Frequency",
          xlab = "No. of R-stone hits", main = title)
   }
```

The resulting plot is shown in Figure 2.2. Let's take a moment to understand what this code does before we go any further.

1. We are going to plot four different histograms, corresponding to the four values $i = 15, 25, 100, 1000$. We need to instruct R to plot a 2×2 plot. The code for this would be:

```
> par(mfrow = c(2, 2), pty = "s")
```

This command seems rather obscure, so it might be a good idea to write
an easier-to-remember function that does the same thing (we will use this
function in future):

```
> multiplot <- function(row, col) {
        par(mfrow = c(row, col), pty = "s")
    }
```

2. The next few lines are just fixed values for the simulations; p is the prob-
   ability (0.5) of a R-stone hit; drops is the number of drops in each exper-
   iment (in our example, 40); and replicates is the number of times we
   repeat the experiment.

```
> p <- 0.5
> drops <- 40
> replicates <- c(15, 25, 100, 1000)
```

3. A *for*-loop then begins by setting the variable *i* to the first item in the vec-
   tor replicates (15), and then runs *i* experiments involving 40 raindrops
   each, returning the number of Right-stone hits in each of the *i* runs. Then
   the hist function is used to plot the distribution of these *i* R-stone hits;
   this distribution is stored in the vector called results. Then the loop be-
   gins again, but this time with the second value in the observations vector
   (25) as the value for *i*; then the above procedure is repeated. Eventually,
   the final value in the observations vector (1000) is used as the value for *i*.

The stabilization about a central value that you see in Figure 2.2 is typ-
ical of random phenomena. The central value here is 20. In our coin-tossing
example, recall that we said that 'intuitively, if the probability of a success
is 0.5, then this should be reflected in the *proportion* of successes we see in
our observations.' And we saw that, the more replicates we performed, the
more closely this proportion approached a limiting number, the probability.
In exactly the same way, the proportion of 20 successes, the proportion of
21, of 22 etc., in our current simulation, all approach fixed numbers. We can
get a rough sense of what these are by plotting the RELATIVE FREQUENCY
HISTOGRAM of our previous results (Figure 2.3):

```
> hist(results, breaks = 0:40, freq = FALSE)
```

By inspection it appears that the relative frequency of the most common
result is a little over 0.12 (recall how seldom the exact count 20 occurred in
our initial simulations—we now see it occurs a little over 10% of the time.)
The relative frequencies fall away in a symmetrical pattern and seem to be-
come effectively zero for counts below 10 and above 30. Where does this
pattern come from? Can we characterize it exactly? A common definition

No. Obs.
15

No. Obs.
25

No. Obs.
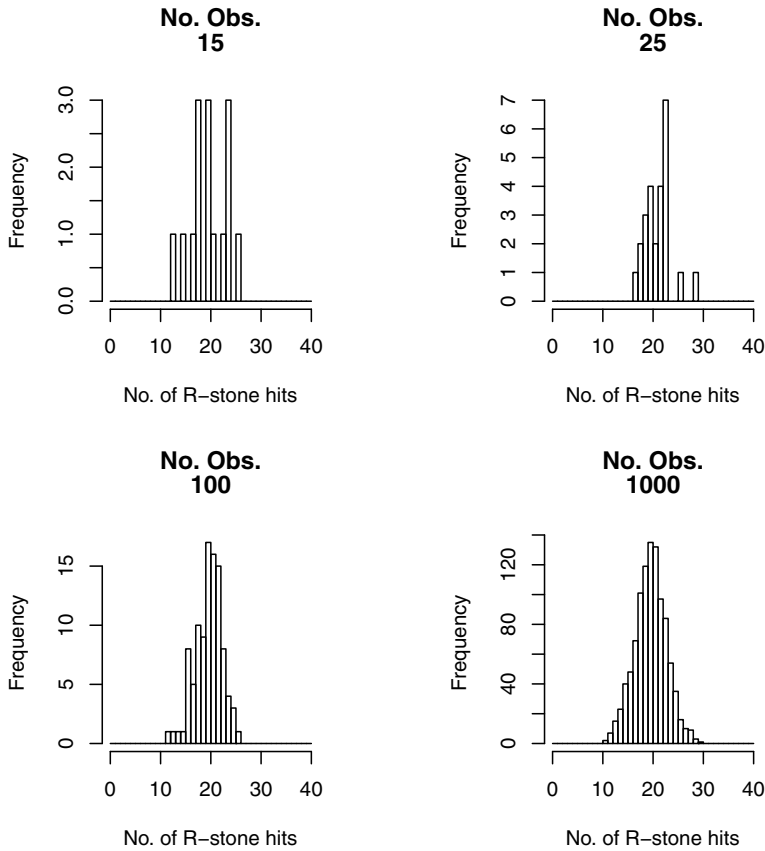**100**

No. Obs.
1000

Fig. 2.2 The frequency of Right-stone hits as the number of replicates increases from 15 to 1000. Note that as the number of replicates increases, the most frequently occurring number of Right-stone hits is in the range of 20—exactly half the total number of drops observed each time.

of probability is the limiting final value of relative frequency. Can we compute the exact probability of each of these counts, without doing an infinite number of simulations? It turns out we can, using our elementary sum and product rules, but the computation is intensive for large numbers (which is why we have R). In order to do this 'by hand' let's focus on the case where the number of trials is just 4.

We have seen that when we simulate a random process, we cannot predict what any particular outcome will be: LRRL, RLLL, RRLL, etc. But we have also seen that not all of these outcomes are equal. If we assign each outcome to its 'success count' (2, 1, 2 etc.), we saw that over many repetitions, some of these counts turn up more frequently than others. Why is this, and what
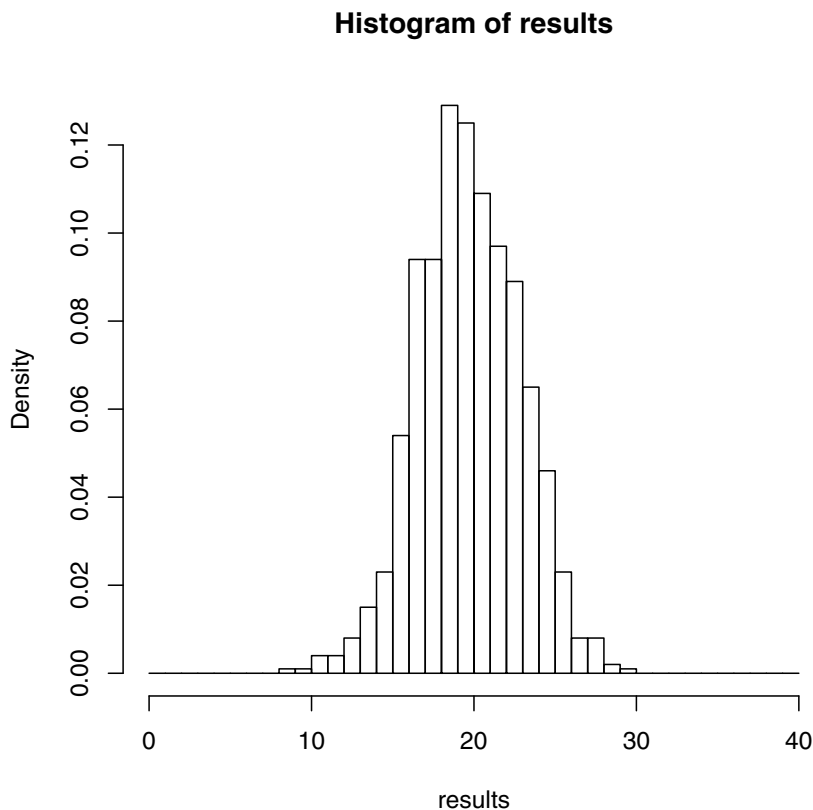
Histogram of results

Fig. 2.3 The relative frequency histogram of the result of observing 40 drops 1000 times.

are the exact proportions? There are five possible such counts: 0, 1, 2, 3, 4. What is the exact probability of each?

Let's focus on one particular count. What is the probability of the event E = 'count is 3'? This is actually a complex event, made up of both sums and products of primitive probabilities. Suppose that in one outcome or event E_1 we see RRRL (in that order). That is:

$$E_1 = (RRRL) \tag{2.3}$$

What's the probability of this happening? Well, we know that $P(L) = P(R) = \frac{1}{2}$, and we know the multiplication rule for independent events.

$$P(E_1) = P(R) \times P(R) \times P(R) \times P(L) = \frac{1}{16} \tag{2.4}$$

But there are four distinct outcomes that yield three Rights-stone hits, call them E_1, E_2, E_3, E_4: E_1 = RRRL, E_2 = RRLR, E_3 = RLRR, E_4 = LRRR. So we have a complex event E, made up of four mutually exclusive possibilities:

$$E = E_1 \text{ or } E_2 \text{ or } E_3 \text{ or } E_4 \tag{2.5}$$

which means we can use the summation rule:

$$P(E) = P(E_1) + P(E_2) + P(E_3) + P(E_4) = \frac{1}{4} \tag{2.6}$$

You can already see that figuring out the answer is going to be a pretty tedious business. Let's think of a better way to work this out. Consider the visualization in Figure 2.4 of the PROBABILITY SPACE when we carry out four trials.
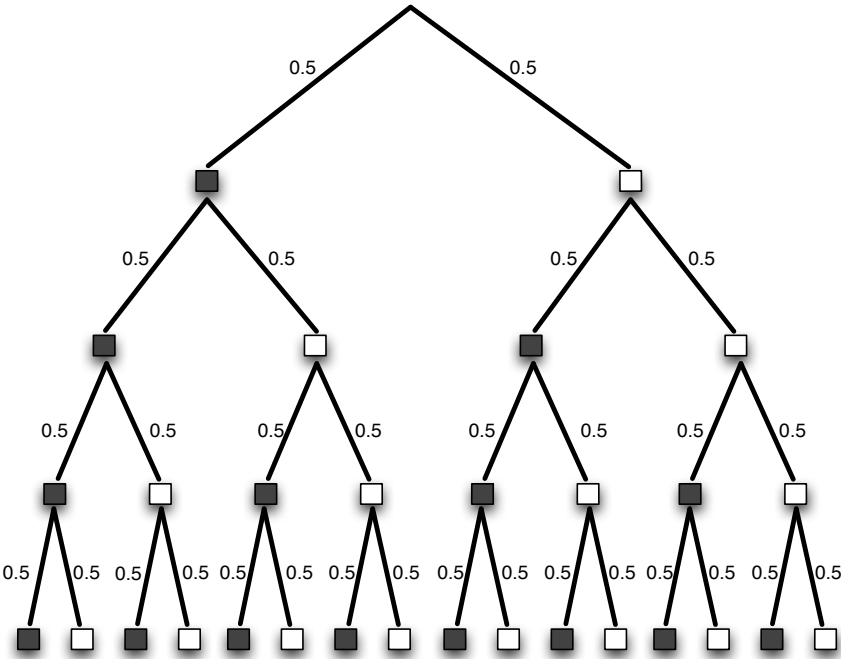

Fig. 2.4 The probability space for four raindrops falling on Left (black) and Right (white) stones.

Figure 2.4 helps us work out the relevant answers. The top of the tree represents the initial state of the probability space, when no raindrop has been observed. When we observe a raindrop once, we can get either a Left-stone hit or a Right-stone hit, and these mutually exclusive events are represented

by the two boxes (black for a Left-stone hit and white for a Right-stone hit). Each is an equiprobable event. After one of these possible events, another trial will yield either a Left-stone hit or a Right-stone hit, and so on for the other two trials. So if we go from the top to the bottom of this tree, following each possible path, we have all the logically possible outcomes of Left- and Right-stone hits in this four-trial example. Thus we get: probability of zero Right-stone hits: 0.5^4; probability of only one Right-stone hit: 4×0.5^4; probability of exactly two Right-stone hits: 6×0.5^4; probability of three Right-stone hits: 4×0.5^4; and probability of four Right-stone hits: 0.5^4.

If we multiply the probabilities along each path of the tree and then add them up, they will sum to 1, since these exhaust the possibilities. This visualization method generalizes to any number of trials, and to experiments involving more than two possible outcomes (e.g., the toss of a die).

2.2 The Binomial Distribution

An essential part of this computation then, is knowing *how many* outcomes yield the same count: how many ways are there to have three R's and one L, say. It turns out there are 4 outcomes of count 3 (as we worked out by hand). All such outcomes have the same probability, $\frac{1}{16}$, so the probability of this count will be $\frac{1}{16}$ multiplied by a factor of 4. This factor is well known from combinatorial theory. We want the number of ways we can arrange 3 R's in 4 positions. This is $\binom{4}{3}$ (read "four choose three"), known as the BINOMIAL COEFFICIENT (its formula can be found in any text on discrete math or combinatorial theory such as Rosen, 2006). It is available as a built-in function in R:

```
> choose(4, 3)
```

```
[1] 4
```

The number of ways of arranging $0 \ldots 4$ R's in 4 positions is:

```
> choose(4, 0:4)
```

```
[1] 1 4 6 4 1
```

(The 'positions' here are the physical arrangement of trial stones on the pavement, or the temporal sequence of trials using one pair of stones). Returning to our larger example, the number of possible 13 Right-stone hits in 40 trials is:

```
> choose(40, 13)
```

```
[1] 12033222880
```

And we can display the number of ways of arranging 0:40 R's in 40 positions (Figure 2.5):

```
> num.outcomes <- choose(40, 0:40)
> plot(num.outcomes)
```
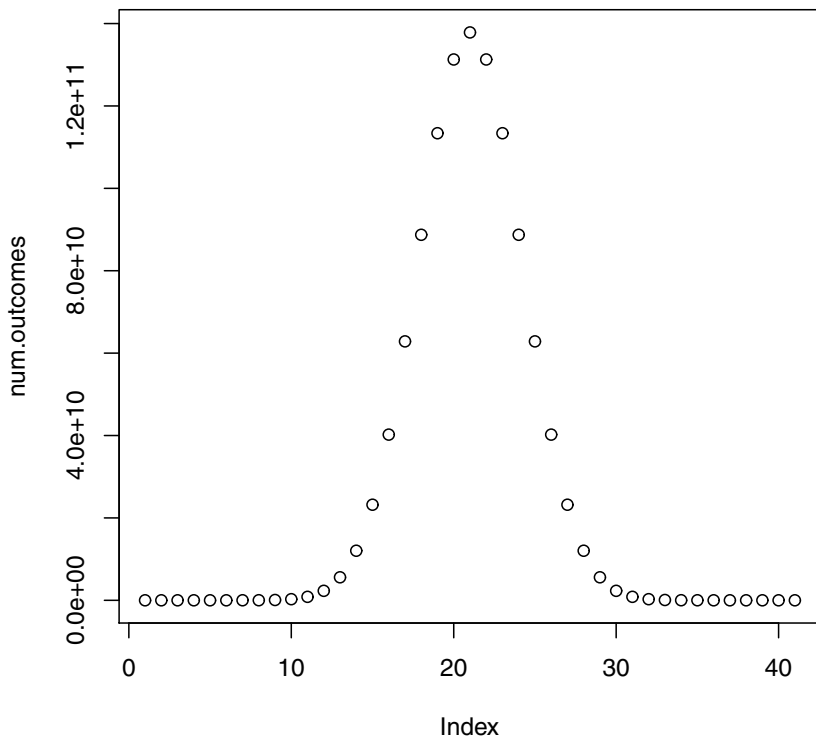

Fig. 2.5 The binomial coefficient of 0:40 Right-stone hits.

To get the exact probabilities of the counts, we just need to multiply the number of outcomes corresponding to each count by each outcome's individual probability. Assuming equiprobable primitive events, which simplifies things a little, this will be, say, $(0.5^{19} \times 0.5^{21})$ for the count 19, $(0.5^{20} \times 0.5^{20})$ for the count 20, etc., which are all equal to each other, and all equal to 0.5^{40}.

```
> p <- (0.5^40)
[1] 9.094947e-13
```

Note the minuscule probability of any particular outcome! This is the probability of getting the *exact* sequence LRLLRRLRLR...RLR of length 40, for example. Finally we combine this number with the binomial coefficients to produce the exact probabilities for each count (Figure 2.6).

```
> binom.distr.40 <- choose(40, 0:40) * p
> plot(binom.distr.40)
```
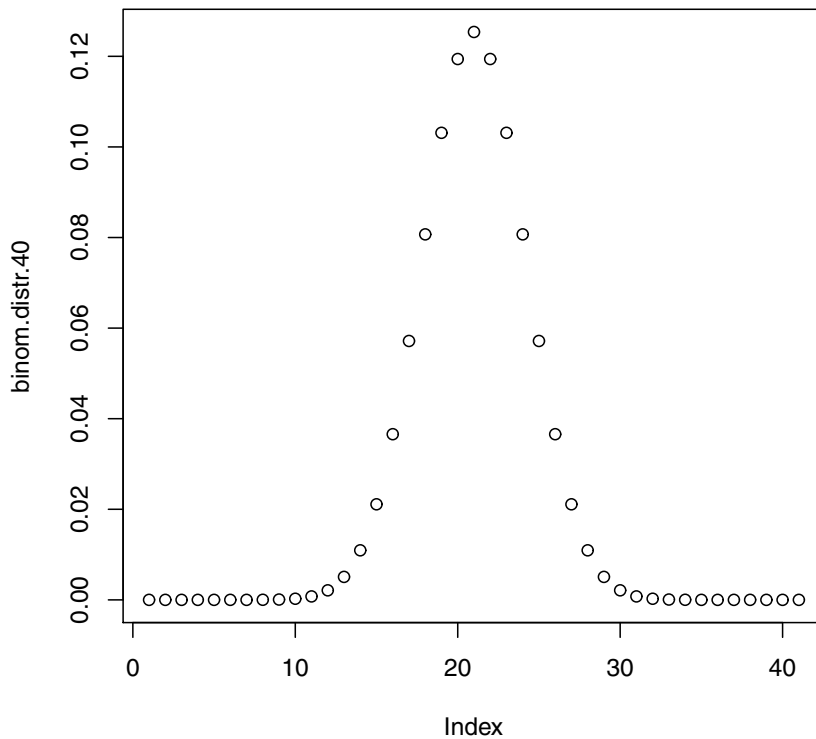

Fig. 2.6 The binomial distribution for $n = 40$.

Since these are all of the mutually exclusive possibilities, the sum of their probabilities should equal 1.

```
> sum(binom.distr.40)
```

```
[1] 1
```

And now we can compute the *exact* probability of observing exactly 20 hits:

```
> choose(40, 20) * p
```

```
[1] 0.1253707
```

To summarize, the BINOMIAL THEOREM allows us to compute the probability of k Right-stone hits (successes) when we make n observations (trials), when the probability of a Right-stone hit (success) is p:

$$\binom{n}{k} \times p^k (1-p)^{n-k} \tag{2.7}$$

The binomial theorem can be applied whenever there are only two possible primitive outcomes, the fixed, n trials are mutually independent, and the probability p of a 'success' is the same for each trial.

Recall the pattern that emerged as we simulated raindrops: as the number of replicates (the number of times we observe 40-drop sequences) increases from 15 to 1000, the *relative frequencies* of R-stone hits settle down to stable values. The *distribution* of R-stone hits has a stable final shape. Just as we express each final *value* in terms of a *theoretical probability*, so we can express the final *shape* in terms of a *theoretical probability distribution* (which we approached empirically using simulation and have now derived mathematically from the primitive probabilities). This shows us how the total probability is distributed among all the possible results of an experiment

The 'central value' of the distribution is in fact the MEAN of the distribution:

```
> mean(results)
```

```
[1] 20.181
```

And as the number of replicates increases, the closer this mean approaches the theoretical limit of 20. The **mean of the sampling distribution** will be explored in great detail as we go forward.

2.3 Balls in a Box

Let's now exercise our new-found knowledge about the binomial distribution in a different scenario, and formally introduce the concepts of sampling theory.

Suppose we have 12,000 balls in a big box, and we *know* that 9000 (i.e., 3/4) are Red, the others White. We say we have a POPULATION of 12,000. Suppose we take a RANDOM SAMPLE of 100 balls from these 12,000. We'd expect to draw about 75 white balls. What's the probability of getting *exactly* 75?

We can simulate this scenario by repeatedly drawing random samples from such a box. For each sample we compute a number, the count of Red balls or 'successes.' A number that describes some aspect of a sample is called a STATISTIC. The particular statistic we are computing here is the SAMPLE COUNT, and if we plot the results we will be able to get an idea of the SAMPLING DISTRIBUTION of this statistic.

The result of the simulation is shown in Figure 2.7, and once again it's worth pausing to examine the code (shown below). In this simulation, and in many others we will encounter, the `for`-loop steps through the replicates, and for each replicate the result is added to a vector, here `sample.counts`. This vector needs to be *initialized* before we can add values to it: this happens prior to the `for`-loop, where we fill the vector with 1000 missing values, symbolized by `NA` in R, using the `rep()` function (short for 'repeat'). We use the same function to fill the box with 9,000 Reds and 3,000 Whites. We code Red as 1, and White as 0, purely for convenience, since then we can compute the sample count of Reds simply by summing the sample itself. The function `sample()` is used to extract a sample of the specified size. Finally, we plot the frequency and relative frequency histogram of the sample counts.

```
> box <- c(rep(1, 9000), rep(0, 3000))
> sample.counts <- rep(NA, 1000)
> for (i in 1:1000) {
      a.sample <- sample(box, 100)
      sample.counts[i] <- sum(a.sample)
  }
> multiplot(1, 2)
> hist(sample.counts, breaks = 50:100)
> hist(sample.counts, breaks = 50:100, freq = FALSE)
```

Inspection of the distribution shows that the central and most probable value is indeed close to 75, with a relative frequency of about 0.09. The pattern approximates a binomial distribution. Can we compute the exact probability of obtaining a sample with exactly 75 balls? As we now know, this partly depends on *how many ways* it is possible to draw a sample whose count equals 75 Reds. We need the binomial theorem:

$$\binom{n}{k} \times p^k (1-p)^{n-k} \tag{2.8}$$

Let's first define a function in R to calculate this quickly:

```
> binomial.probability <- function(k, n, p) {
      choose(n, k) * p^k * (1 - p)^(n - k)
  }
```

If we run that function now with $k = 75$, $n = 100$, $p = 3/4$, we find that the probability of getting *exactly* 75 is:
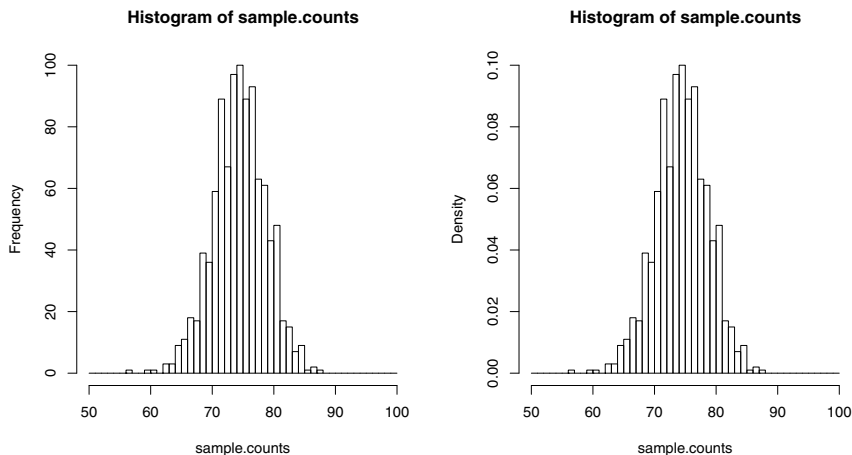
Fig. 2.7 The sampling distribution for the sample count of Red balls in a sample of 100, showing absolute frequency (on the left) and relative frequency (on the right).

```
> binomial.probability(75, 100, 3/4)
```

```
[1] 0.0917997
```

As an aside, note that R provides this function under the name of `dbinom`:

```
> dbinom(75, 100, 3/4)
```

```
[1] 0.0917997
```

Now we ask an interesting question: suppose the sample was larger, say 1000. Intuitively, by increasing the sample size, we should get a better estimate of what is actually in the box. After all, if the sample size was $12,000$, we would *know* the contents of the box, and the larger our sample, the closer we should get to certainty. Would the probability of drawing 3/4 Red balls (750) from this larger sample be higher or lower than the number (0.0918) that we computed for a sample 10 times smaller? Think about this before reading further...

Here is the answer:

```
> binomial.probability(750, 1000, 3/4)
```

```
[1] 0.02912411
```

Thus, as the sample size goes up, the probability of the most likely sample count goes *down*. And this goes for the probabilities of *all* the counts in the larger sample. The reason for this is that a total probability of 1 is being

distributed among 101 possibilities in the first case, but among 1001 possibilities in the second. The probability is 'stretched thin' in the larger sample. 750 Reds is still the *most probable* sample count, but in absolute terms, its probability decreases. This leads us to a fundamental principle of statistical inference: we are hardly ever interested in the probability of a *single value.* Instead, we focus on a *range* or INTERVAL of possible values, and compute the probability of being within that range.

We first focus on how to compute the probability of such an interval; we then examine how this probability behaves as we alter the sample size.

To explore this approach, consider an alternative (simpler) scenario where we have 12,000 Red and White balls, and exactly half are Red ($p = 0.5$). We take a sample of 40 balls, and calculate the probability of getting $1\ldots 40$ Reds:

```
> n <- 40
> p <- 0.5
> probs <- binomial.probability(0:n, n, p)
```

This function simply computes `binomial.probability(0, 40, 0.5)`, `binomial.probability(1, 40, 0.5)`, `binomial.probability(2, 40, 0.5)`, etc., and puts each probability in the vector `probs`. In this and all the code in the rest of this chapter, `n` represents the size of the sample we draw.

The variable `probs` now contains a vector of probabilities. (Note that the probability of the count 20 is indexed as `probs[21]`. This is because the probability corresponding to 0 Reds is in `probs[1]`. There is no such thing as `probs[0]`. This kind of offset is a commonly encountered in computer coding). Here are the central values of this vector.

```
> probs[20:22]
```

```
[1] 0.1194007 0.1253707 0.1194007
```

The probability of getting exactly 20 Red balls is 0.1254. As we have come to expect, this probability is quite low. Recall too that the value 20 is the mean of the sampling distribution. Let's investigate the probability of an interval around this value.

What's the probability of getting—not exactly 20—but 19 or 20 or 21 Red balls in a sample of 40? This is an interval centered on 20, with a MARGIN OF ERROR ± 1. Since these are three mutually exclusive alternatives, we can use the sum rule:

```
> sum(probs[20:22])
```

```
[1] 0.364172
```

And for margin of error $\pm$ 2:

```
> sum(probs[19:23])
```

```
[1] 0.5704095
```

Let's go ahead and compute the probabilities for *all* the margins of error.

```
> mean.index <- 21
> intervals <- rep(NA, 20)
> for (i in 1:20) {
       indices <- seq(mean.index - i, mean.index +
           i, by = 1)
       intervals[i] <- sum(probs[indices])
   }
> conf.intervals <- data.frame(margin = rep(1:20),
       probability = intervals)
> head(conf.intervals)

  margin probability
1      1   0.3641720
2      2   0.5704095
3      3   0.7318127
4      4   0.8461401
5      5   0.9193095
6      6   0.9615227
```

The important point here is that when we increase the margin of error to be ±6 around the precise expected mean number of Red balls (20), the probability of *a value within this interval* occurring is approximately 0.95. Let's visualize this (Figure 2.8).

```
> main.title <- "Sample size 40"
> plot(conf.intervals$margin, conf.intervals$probability,
       type = "b", xlab = "Margins", ylab = "Probability",
       main = main.title)
> segments(0, 0.95, 5.7, 0.95)
> segments(5.7, 0, 5.7, 0.95)
```

The straight lines in Figure 2.8 mark the margin of error (about 6), which corresponds to 0.95 probability. What this means is, as we take repeated samples from the box, the sample count will be *exactly* 20—the exact mean of the distribution—only about 12% of the time, but within 6 of the mean about 95% of the time. Notice too how quickly the curve rises to 0.95. This is due to the simple but important fact that most of the probability is clustered around the mean in the binomial distribution. So a relatively small interval around the mean captures most of the probability.

It is now time to take one of the most important conceptual steps in statistical inference: a kind of inversion of perspective. *If the sample count is within 6 of the mean 95% of the time, then 95% of the time the mean is within 6 of the sample count.* We can use this fact to infer the mean of the
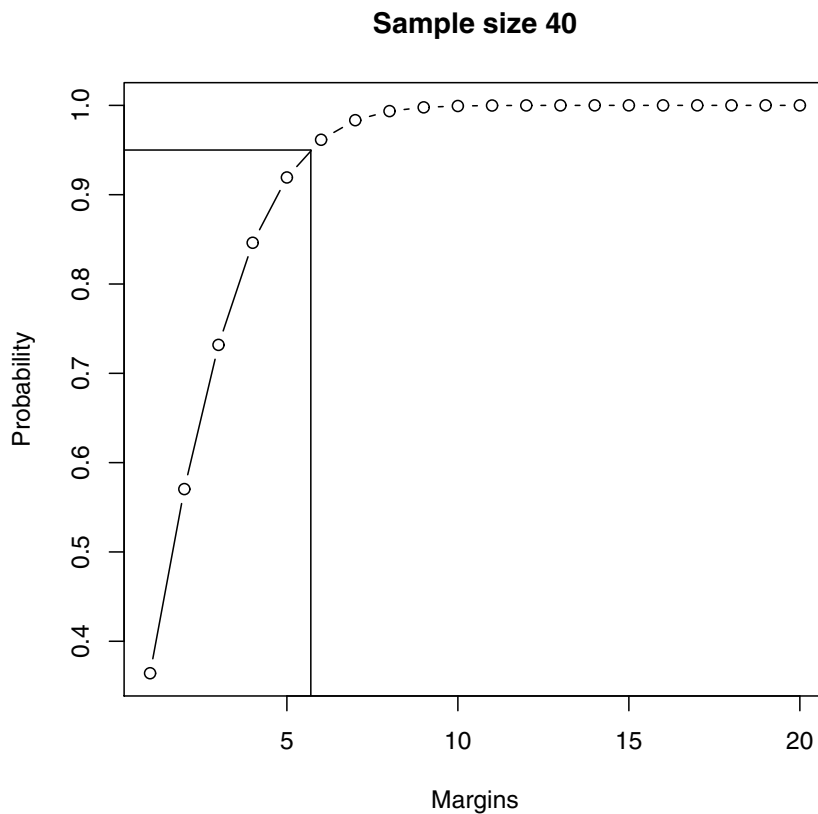
Sample size 40

Fig. 2.8 The probability of getting $20 \pm m$ Red balls in a random sample of 40, where m is the margin of error we allow (ranging from 1 to 20).

distribution from a single sample (however, see page 57 for a subtle point relating to this inversion). We cannot know *exactly* what it is, but we can express a 95% LEVEL OF CONFIDENCE that it is in an interval within 6 of the sample count. To take a concrete example:

```
> box = c(rep(1, 6000), rep(0, 6000))
> sum(sample(box, 40))

[1] 17
```

We do not know what specific value we will get when we execute this code (random processes are unpredictable in specific cases). But exploiting our knowledge of the pattern of random phenomena in the aggregate, we *know* that the value is within 6 of 20, 95% of the time. You may wish to

execute the above code multiple times to prove this to yourself. Note the logic: we first construct the CONFIDENCE INTERVAL around the mean of the sampling distribution—this tells us how large the interval must be to capture the specified amount of the probablity. We then take a single sample, compute the SAMPLE STATISTIC (the sample count), and center the interval *on the sample*. This gives us an interval estimate of the value of the mean of the distribution.

We have constructed a 95% confidence interval above. The figure 95% is an arbitrary but conventional standard in statistical inference. We could choose to construct a 99% interval, and if we did, we would have to increase the size of the interval to capture 0.99 of the probability. Our confidence that this interval contains the mean, if centered on any individual sample, would increase, but the *accuracy* of the interval is lower. This is a necessary tradeoff. What we would like to achieve is both high confidence *and* high accuracy. How can we achieve this?

What would happen if the sample size were increased from 40 to 400? Our expected mean number of Red balls would now be 200. The confidence intervals are computed in the following code, and graphed in Figure 2.9.

```
> n <- 400
> p <- 0.5
> probs <- binomial.probability(0:n, n, p)
> mean.index <- 201
> intervals <- rep(NA, 200)
> for (i in 1:200) {
      indices <- seq(mean.index - i, mean.index +
          i, by = 1)
      intervals[i] <- sum(probs[indices])
  }
> conf.intervals <- data.frame(margin = rep(1:200),
      probability = intervals)
> head(conf.intervals)

  margin probability
1      1   0.1192112
2      2   0.1973747
3      3   0.2736131
4      4   0.3472354
5      5   0.4176255
6      6   0.4842569

> main.title <- "Sample size 400"
> plot(conf.intervals$margin, conf.intervals$probability,
      type = "b", xlab = "Margins", ylab = "Probability",
      main = main.title)
> segments(-6, 0.95, 19, 0.95)
> segments(19, 0, 19, 0.95)
```

Inspection of Figure 2.9 and the full list of intervals computed in the code show that the 95% margin of error in this case is about ± 19.
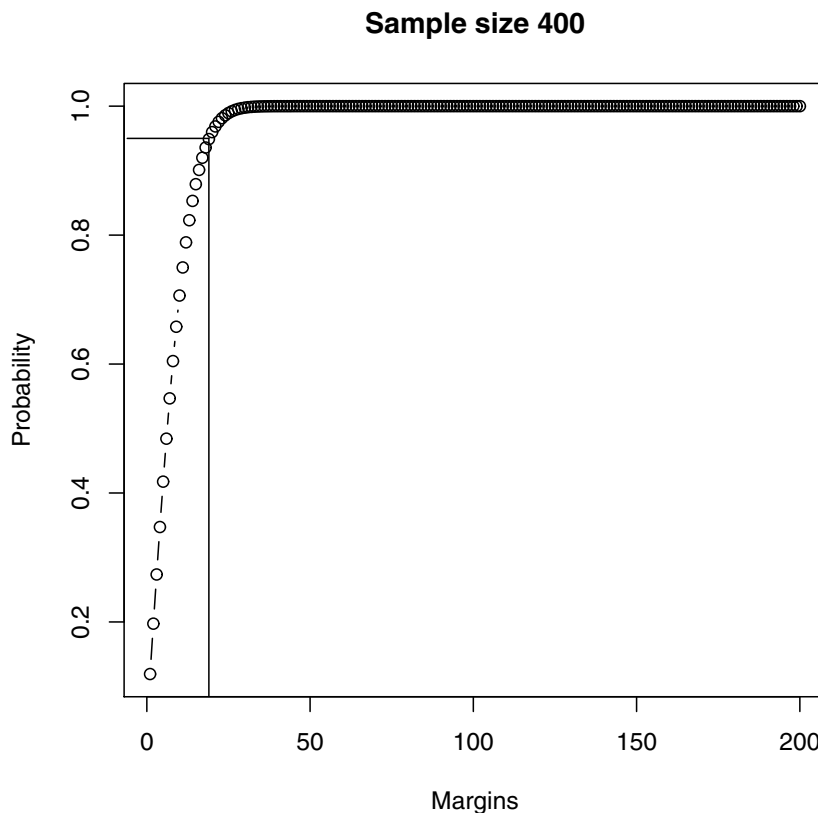
Sample size 400

Fig. 2.9 The probability of getting $200 \pm m$ Red balls from a random sample of 400, where m is the margin of error we allow (ranging from 1 to 200).

(In the above examples, we essentially used the same R code twice, with just a few changes. In such situations it makes sense to write a function that can do the same thing, but with different settings—here, different sample sizes. Let's write such a function. The explanation for each computation is shown in the source code at the course website, but try to work it out yourself first).

```
> compute.margins <- function(sample.size, p) {
      probs <- binomial.probability(0:sample.size,
          sample.size, p)
```

```
    mean.index <- (sample.size * p) + 1
    max.margin <- sample.size * p
    intervals <- rep(NA, max.margin)
    for (i in 1:max.margin) {
        indices <- seq(mean.index - i, mean.index +
            i, by = 1)
        intervals[i] <- sum(probs[indices])
    }
    conf.intervals <- data.frame(margin = rep(1:max.margin),
        probability = intervals)
    return(conf.intervals)
}
```

What then is the effect of sample size on our 95% confidence interval?
What we have established is that as the sample size increases, so does the
confidence interval, *in absolute terms*. For sample size 40, the margin of error
is ±6; for a sample of 400, the margin of error is ±19. The accuracy of our
estimate seems to have become less precise! In order to properly compare the
margins of error, however, we need to NORMALIZE them so that their range is
comparable in both cases (currently, in the 40 sample case the margins range
from 1 to 20, and in the 400 sample case they range from 1 to 200). This
normalization can be done by converting them to proportions. For example,
in the 40 sample case, we simply treat the margin ±1 (19 and 21) as 19/40
and 21/40 respectively; for the 400 sample case, we treat the margin ±1 (199
and 201) as 199/400 and 201/400 respectively. When we do this, the result
is Figure 2.10. (We first define a function `plot.margins()` to display the
margin of error).

```
> plot.margins <- function(sample.size, p, color = "black",
      margin, main, interval = TRUE) {
      probs <- binomial.probability(0:sample.size,
          sample.size, p)
      proportions <- 0:sample.size/sample.size
      plot(proportions, probs, type = "l", col = "black",
          xlab = "Proportions", ylab = "Probability",
          main = main)
      if (interval == TRUE) {
          segments(proportions[(sample.size/2 +
              1) - margin], -0.5, proportions[(sample.size/2 +
              1) - margin], 0.06, col = color,
              lty = 1, lwd = 2)
          segments(proportions[(sample.size/2 +
              1) + margin], -0.5, proportions[(sample.size/2 +
              1) + margin], 0.06, col = color,
              lty = 1, lwd = 2)
```

```
    }
  }
```

Then we plot the margins (Figure 2.10):

```
> multiplot(1, 2)
> main.title.40 <- "Sample size 40"
> main.title.400 <- "Sample size 400"
> plot.margins(40, 0.5, margin = 5, main = main.title)
> plot.margins(400, 0.5, margin = 19, main = main.title)
```

There are a number of important insights to take away from Figure 2.10:

1. As sample size increases from 40 to 400, we get proportionally tighter 95% probability regions: the accuracy of the confidence interval increases.
2. We have effectively employed a new statistic: the SAMPLE PROPORTION. The 40 sample count 21, for example, is replaced by 21/40, the 400 sample count 201, is replaced by 201/400. Figure 2.10 actually shows the sampling distribution of this statistic for two different sample sizes. We explore this further in the next section.
3. The mean of the sampling distribution (of the sample proportion) accurately reflects the proportion of balls in the population. Just as a statistic describes some aspect of a sample, a PARAMETER describes some aspect of a *population*. Thus the mean of the sampling distribution 'points to' the POPULATION PARAMETER.
4. Just as the mean of the sampling distribution is of special significance, so is its standard deviation. Intuitively, the smaller the standard deviation, the tighter the confidence interval, and the more accurate the interval estimate of the mean. In fact, we can quantify this exactly. As will become clear in the coming chapters, the 95% probability region corresponds to approximately 2 times the standard deviation (SD) of the sampling distribution. The smaller the SD, the better the estimate.

In the next section we examine exactly how the SD of the sampling distribution varies with sample size.

Aside: Applying the binomial theorem

As mentioned above, when we want to compute the probability of getting 0 to 20 Right-stone hits when we observe 40 raindrops, we can do this using dbinom:

```
> sums <- rep(NA, 21)
> for (i in 0:20) {
      sums[i + 1] <- dbinom(i, 40, 0.5)
  }
> sum(sums)
```
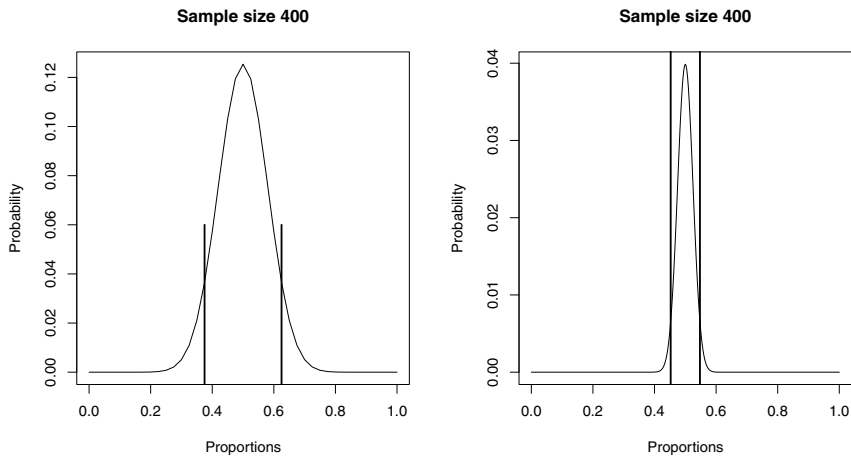
Fig. 2.10 The 95% probability ranges in the 40 and 400 sample case with the margins of error normalized.

```
[1] 0.5626853
```

An even easier way to do this in R is:

```
> sum(dbinom(0:20, 40, 0.5))
```

```
[1] 0.5626853
```

And yet another way is to say:

```
> pbinom(20, 40, 0.5)
```

```
[1] 0.5626853
```

Thus, there is a family of functions for the binomial distribution that we can use to do very useful things:

1. rbinom: the random number generation function
2. dbinom: The probability density function
3. pbinom: The cumulative distribution function (the proportion of values which have a value x or lower)

2.4 Standard Deviation and Sample Size

Earlier we looked at the sampling distribution of the sample count using a
40-drop sequence. Suppose we plot the result of 100 replicates of observing *n*
drops, where *n* is (a) 4, (b) 40, (c) 400, and (d) 4000 (Figure 2.11).

```
> p <- 0.5
> sample.sizes <- c(4, 40, 400, 4000)
> multiplot(2, 2)
> xlabel <- "Number of R-stone hits"
> for (n in sample.sizes) {
      results <- rbinom(100, n, p)
      maintitle <- paste(n, "drops", sep = " ")
      hist(results, xlim = range(c(0:n)), xlab = xlabel,
          main = maintitle)
  }
```

As we increase the sample size from 4 to 4000 (and observe these *n*-drop
sequences 100 times), the spread, i.e., the standard deviation, appears to
decrease. We can see this visually in Figure 2.11. But does it really? Let's
plot the standard deviation by sample size, examining all sample sizes from
1 to 100 (Figure 2.12).

```
> sample.sizes <- 1:400
> p <- 0.5
> standard.deviations <- rep(NA, 400)
> for (n in sample.sizes) {
      binomial.distribution.count <- rbinom(100,
          n, p)
      standard.deviations[n] <- sd(binomial.distribution.count)
  }
> plot(sample.sizes, standard.deviations, xlim = c(1,
      400), xlab = "Sample size", ylab = "Standard deviation")
```

To our great surprise, we get increasing values for standard deviation as
sample size increases. Look carefully at the scale on the *x*-axis in each plot.
In the first it ranges from 0–4, in the last it ranges from 0–4000. What's
going on is that in *absolute* terms the standard deviation of the sample count
increases, but as we have seen, raw counts are not directly comparable; we
need to relativize the count to the different sample sizes.

If we look at sample proportions rather than sample counts, we'll have
normalized the various counts so that they are comparable. So, when we take
a sample of 40 drops, instead of saying 'the count is 18 Right-stone hits,'
we say 'the proportion of Right-stone hits is 18/40.' Let's plot by sample
proportion rather than sample count and see what we get. At this juncture
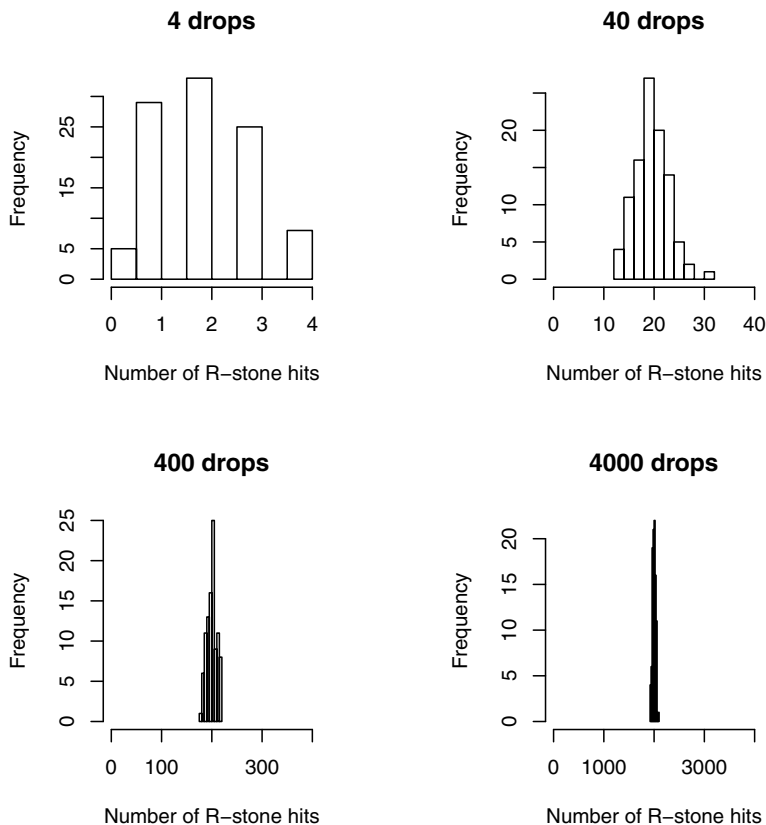you should spend a few minutes trying to modify the previous code in order to

4 drops

40 drops

400 drops

4000 drops

Fig. 2.11 Does increasing the number of drops observed from 4 to 4000 results in a tighter distribution?

plot sample proportions rather than sample counts. The results are shown in (Figure 2.13). Now let's plot the standard deviation of the proportion-based counts (Figure 2.14).

```
> p <- 0.5
> sample.sizes <- c(4, 40, 400, 4000)
> multiplot(2, 2)
> for (n in sample.sizes) {
      results <- rbinom(100, n, p)/n
      maintitle <- paste(n, "drops", sep = " ")
      xlabel <- "Proportion of R-stone hits"
      hist(results, xlim = range(c(0:1)), xlab = xlabel,
          main = maintitle)
  }
```
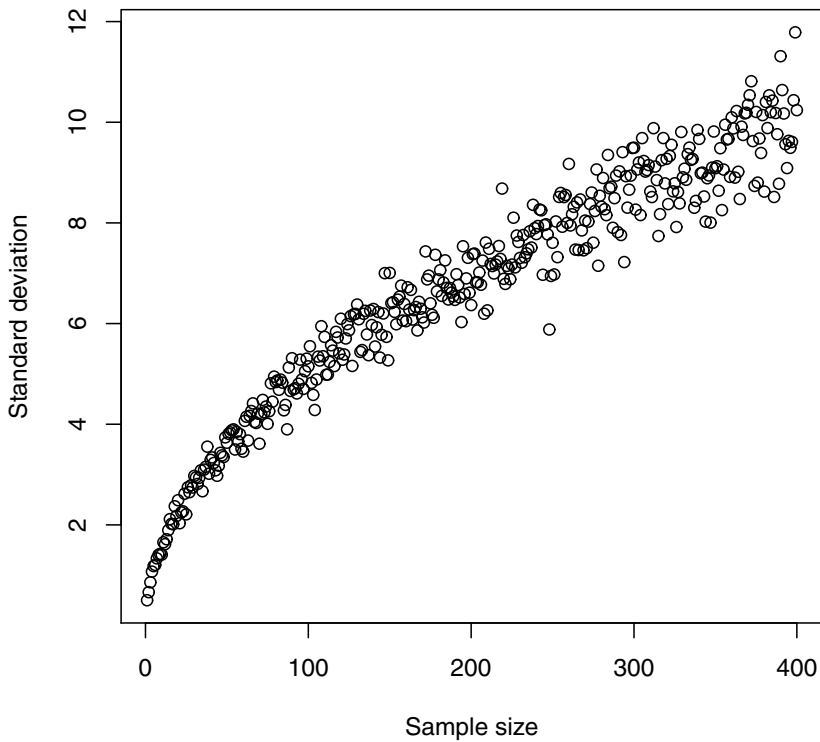
Fig. 2.12 Standard deviation of the sample count increases with sample size.

Plotting the standard deviations of proportions now yields the expected pattern of decreasing standard-deviation values with increasing sample size:

```
> sample.sizes <- 1:400
> p <- 0.5
> standard.deviations <- rep(NA, 400)
> for (n in sample.sizes) {
      binomial.distribution.prop <- rbinom(100,
          n, p)/n
      standard.deviations[n] <- sd(binomial.distribution.prop)
  }
> plot(sample.sizes, standard.deviations, xlim = c(1,
      400), xlab = "Sample size", ylab = "Standard deviation")
```
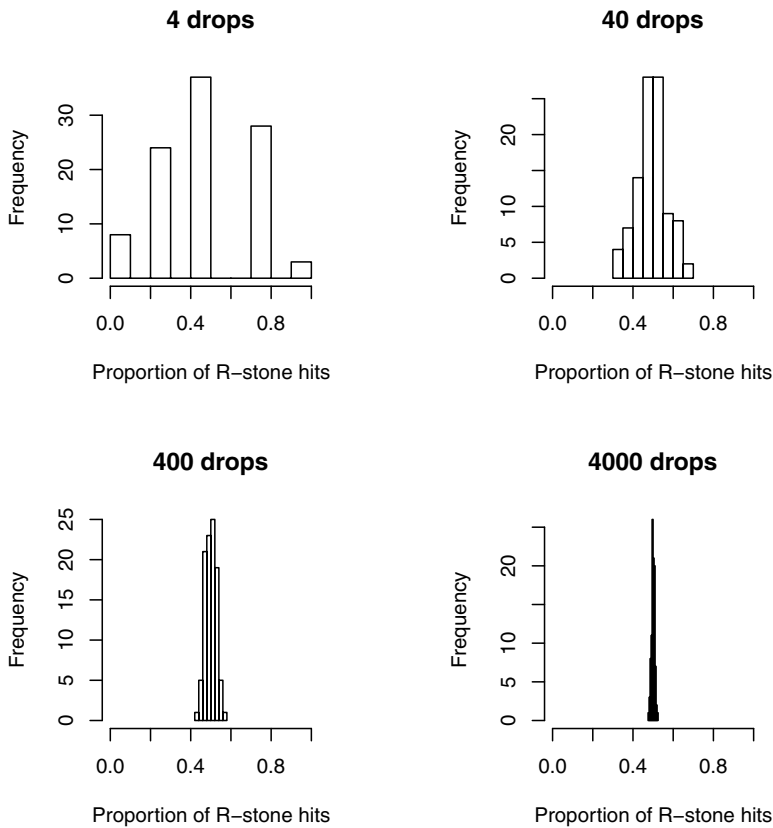
Fig. 2.13 Plot of distribution of sample proportion of Right-stone hits as sample size increases.

Now everything makes sense: look at the scale of the *x*-axis in these plots. The spread, or standard deviation, decreases as we increase sample size. This is an important insight that we will come back to. We will also revisit this technique of normalizing the scale of comparison in coming chapters.

2.4.1 Another Insight: Mean Minimizes Variance

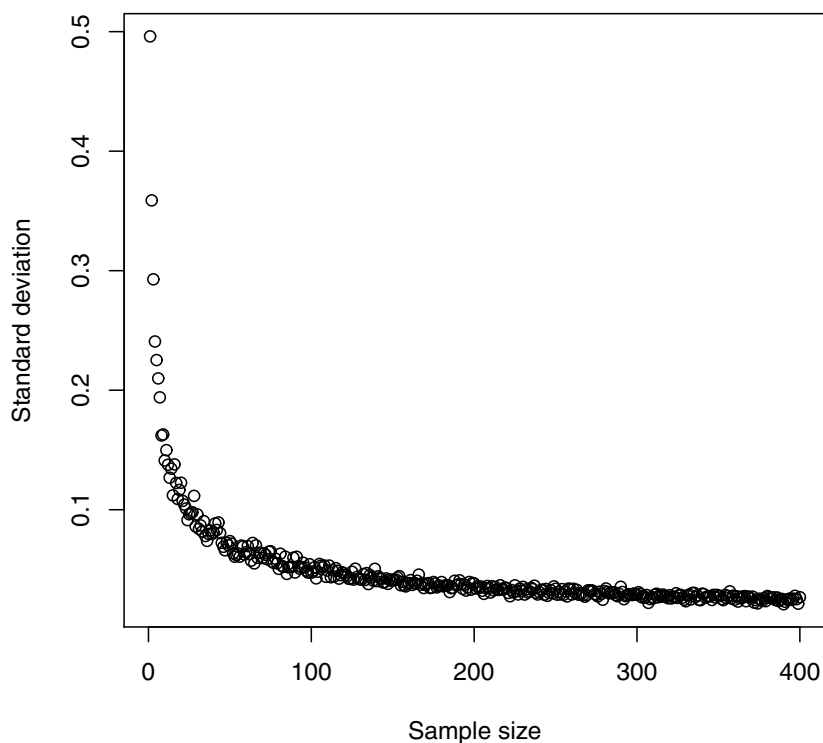Recall that variance is defined as follows:

Fig. 2.14 Standard deviation of the sample proportion decreases as sample size increases.

$$s^2 = \frac{(x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + \cdots + (x_n - \bar{x})^2}{n-1} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \qquad (2.9)$$

Standard deviation, s, is a measure of spread about the mean, as we just saw. Recall our earlier observation that the sum of deviations from the mean will always equal zero.

A related fact is that the SUM OF SQUARED DEVIATIONS from the mean are smaller than from any other number—the mean is a special number in that sense.

The sum of squared deviations from the mean simply means that, given a vector of scores, we calculate the mean of the vector, and then, subtract each value in the vector from the mean, squaring the result each time; and then

we sum up these squared deviations. Consider this simple example; let the vector of scores be a range of values going from 1 to 10:

```
> vector <- 1:10
```

We can compute its mean:

```
> vector.mean <- mean(vector)
```

Then, we can take squared deviations from the mean of each value in the vector, and then sum them up:

```
> sum((vector - vector.mean)^2)
```

```
[1] 82.5
```

We will be using this insight about the sum of squared deviations when we discuss analysis of variance in chapter 5.

Let's quickly convince ourselves that the sum of squared deviations from the mean are smaller than from any other number. First, we generate all the possible squared deviations from the mean and all other possible numbers:

```
> size <- 1
> p <- 0.5
> num.drops <- 4000
> results <- rep(NA, 100)
> for (i in 1:100) {
      results[i] <- sum(rbinom(num.drops, size,
          p))
  }
> mean.results <- mean(results)
> n <- floor(mean.results - 1)
> m <- floor(mean.results + 1)
> xvalues <- c(1:n, mean.results, m:4000)
> all.sq.deviations <- rep(NA, length(xvalues))
> for (i in xvalues) {
      vector.i <- rep(i, 100)
      deviations <- results - vector.i
      sq.deviations <- sum(deviations^2)
      all.sq.deviations[i] <- sq.deviations
  }
```

Next, we plot the sum of squared deviations from the mean against all other numbers (Figure 2.15).

```
> xlabel <- "Potential minimizers of sum of squared deviation"
> plot(xvalues, all.sq.deviations, xlab = xlabel,
      ylab = "Squared Deviation")
> lines(xvalues, all.sq.deviations)
```
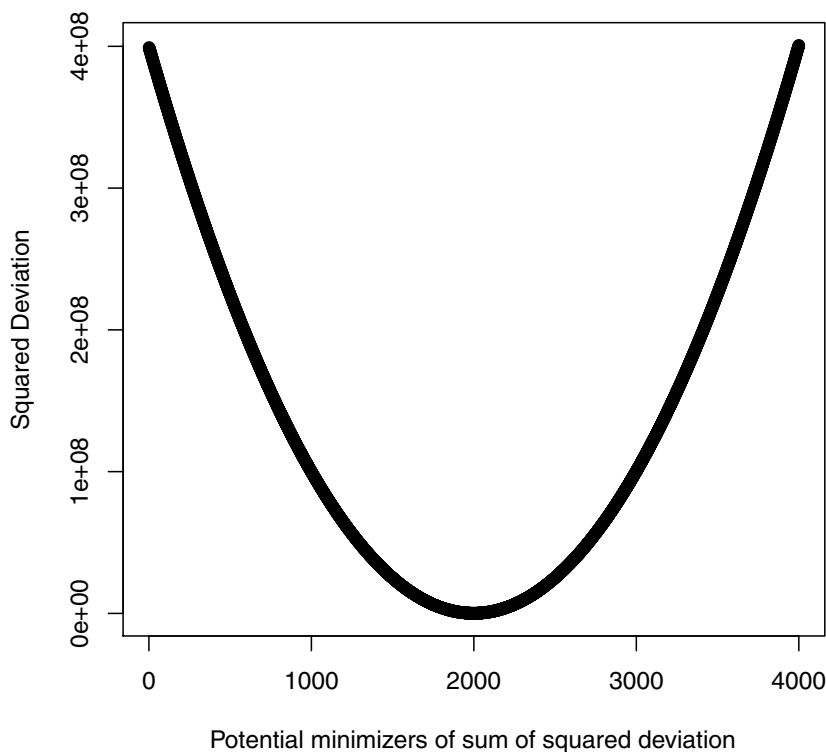
Fig. 2.15 The mean minimizes sum of squared deviations: the sum of squared deviations from the mean are smaller than from any other number.

2.5 The Binomial versus the Normal Distribution

Before we proceed further, we would like to introduce a distribution that is very similar to the binomial distribution shown in Figure 2.6. It is defined by this somewhat intimidating-looking function:

$$f(x) = \frac{1}{(\sigma\sqrt{2\pi})} E^{-((x-\mu)^2/2\sigma^2)} \tag{2.10}$$

Given a range of values for x, and specific values for μ, and σ, we could plot the result of applying this function. Let's define this function, plot it, and compare it to the binomial distribution (Figure 2.16).

```
> new.function <- function(x, mu, sigma) {
      1/(sqrt(2 * pi) * sigma) * exp(1)^(-((x -
          mu)^2/(2 * sigma^2)))
  }

> main.title <- "Comparing the binomial and normal distributions"
> plot.margins(40, 0.5, 40, margin = 20, main = main.title,
      interval = FALSE)
> lines(c(1:40)/40, new.function(c(1:40), 20, 3),
      col = "black", lty = 2)
```
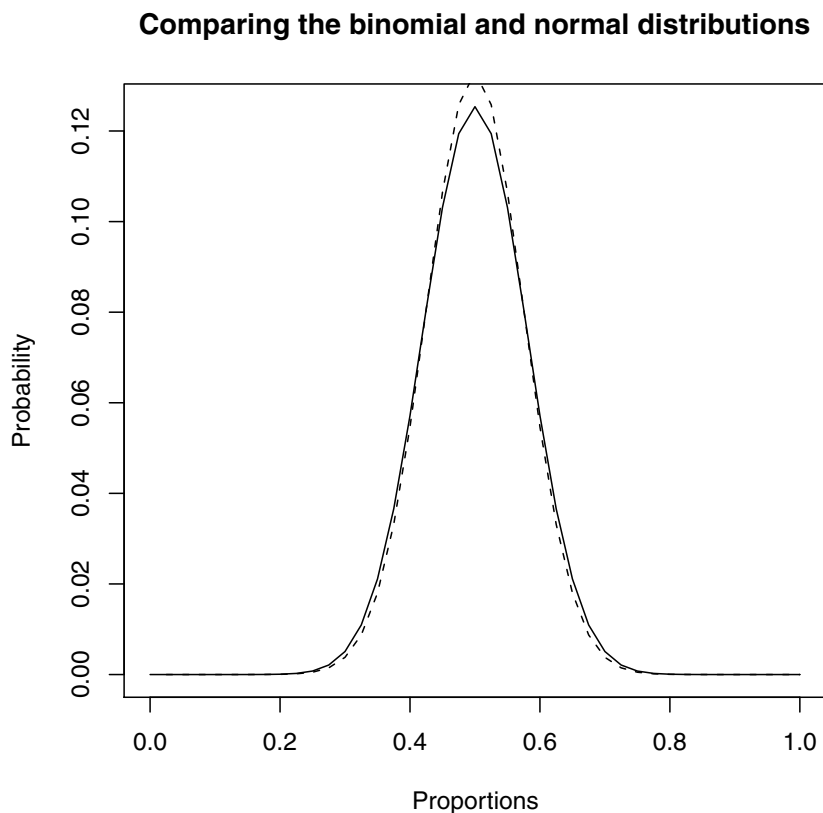

Comparing the binomial and normal distributions

Fig. 2.16 Comparison of the binomial (solid line) and the normal (dashed line) distributions.

This is known as the normal distribution function and has a very similar shape to the binomial distribution. This is not surprising, since the func-

tion was originally formulated by De Moivre in the early 1700s as an *approximation* to the binomial. As we have seen, working with the exact discrete binomial involves fantastically large numbers of sums and products, which were impossible to compute before the advent of digital computers. De Moivre's analytic formulation facilitated proofs of the distribution's mathematical properties.

One important difference between the normal and binomial distributions is that the former refers to continuous dependent variables, whereas the latter refers to a discrete binomial variable. An example of a continuous dependent variable would be reaction time or reading time data. In the next chapter we will use the normal distribution to explore the behavior of a new, continuous statistic: the sample mean.

Problems

2.1. Imagine that you have a biased coin, where the probability of obtaining a heads is not 0.5 but 0.1. When the coin is tossed four times, what are the probabilities of obtaining 0, 1, 2, 3, 4 heads?

2.2. Using a probability-space tree like the one shown in Figure 2.4, calculate the probability of obtaining a 6 each time when a die is tossed three times in a row.

2.3. What is the probability of obtaining any of the numbers 2, 4, or 6 if a die is tossed three times in a row?