Chapter 2

Counting ordinal variables and discrete numerical variables

Contents:

2.1. Big words, small words, and four-letter words
2.2. Ordinal categorical data
2.3. Discrete numerical variables
2.4. Picturing numerical data
2.5. Answering our research question
2.6. Summary
2.7. R code
2.8. Exercises
2.9. References and data sources
Appendix 2A. Some fun facts about word history

Dear Mr. Webster,
My daughter came home from school today, using terrible curse words. I can only assume she learned them from the dictionary, since I never swear. I am shocked that you allow such words in the dictionary.

Dear Madam,
I am sorry to hear of this experience. A dictionary is a tool with many uses. I applaud your daughter’s initiative in looking up unfamiliar words. I wonder, however, how she knew which section to look for them, if she had never heard them before.
2.1. Big words, small words, and four-letter words

There are several English expressions that refer to the “cost” or “size” of a word. At the high end, for example, we have expressions like “four-bit word” and “silver-dollar word,” which, due to inflation, have been upgraded to even bigger denominations such as “twenty-dollar word,” and even “hundred-dollar word.” These expressions refer to generally longer-than-average, more erudite, and sometimes even pretentious sounding words, perhaps like some of the ones in this paragraph. We may also call a word of this sort, simply a “big word,” meaning a type of English word like the the ones in the list of “My Big Words” that we found on the internet by googling “big words”. This list was collected and posted by Carol A. Caronna on her web site. It includes 357 words and short phrases, such as *defenestrate* and *ad hominem.* Caronna links to this list from a section of her site called “Resources for Students,” and suggests that the list can be used “for studying for the GRE, enriching your vocabulary, sounding educated (but don’t use to sound like a snob!)”

Another word-size expression at the other end of the size scale is “four-letter word,” a phrase that the comedian George Carlin used to describe some of the dirty words that you’re not supposed to say over broadcast media. It was a broadcast of this comedic monologue

[1] You can see more examples from this list in Table 1.2, or point your browser to the web page http://pages.towson.edu/ccaronna/vocabulary.htm where we downloaded it on April 4, 2009.

(http://law2.umkc.edu/faculty/projects/ftrials/conlaw/filthywords.html), which led to the FCC’s current standards on regulating “obscene” and “indecent” speech in broadcast. Regarding Carlin’s monologue, “the Commission concluded that certain words depicted sexual and excretory activities in a patently offensive manner”, and the Supreme Court of the United States upheld the FCC’s ability to regulate such language. (See FCC v Pacifica Foundation: https://supreme.justia.com/cases/federal/us/438/726/case.html).

Note how these expressions invoke other perceived characteristics of the words, and not just their size. That is, a four-letter word is not just any word that is four letters long but specifically one that is “profane,” “indecent,” or “obscene.” So, *book, read, home,* and *love* are not “four-letter words” even though they are four letters long. And words that are counted as “four-letter words” may be longer than four letters, as are some of the words in Carlin’s monologue, such as *motherf***er.* Similarly, from the purposes that Caronna gives for posting her list of “My Big Words,” these don’t seem to be just any words that are much longer than four letters. Indeed, the *Oxford English Dictionary* defines the expression “big words” as words that are “intended ... to impress, overawe, or confuse the hearer through being drawn from scholarly or elevated vocabulary.” This intention is what Caronna seems to warn against when she says, “don’t use to sound like a snob!” So, if the opposite of a “big” word is a “small” one, then size here seems to be standing in for some other property, such as the relative obscurity versus familiarity of the
word. Indeed, a sense of familiarity is a critical property that identifies “small words” in an essay written by Richard Lederer, an author who writes about language, and is best known for books on word play such as The Cunning Linguist. In his essay, “The Case for Short Words,” he describes small words as “the ones we seem to have known from the time we were born.”

We can see even more clearly that it’s not just size that makes a word “big” when we look more closely at Caronna’s list. The list includes the words aver, onus, rive, and wily, each of which is only four letters long, but it doesn’t include words such as caterpillar, refrigerator, strawberries, and kindergarten, perhaps because they feel as familiar as many of the words that Lederer uses to argue that: “A lot of small words, more than you might think, can meet your needs with a strength, grace, and charm that large words do not have.” Lederer makes a case for this statement by using only words with one syllable in the title and first four paragraphs of the essay. However, while every word that he uses in this stretch of text is exactly one syllable, a large proportion of them (49 out of 138) are bigger in number of letters than those four smallest “big words” on Caronna’s list. For example, Lederer uses tongue, things, strange, and strength, all of which have more letters than aver, onus, rive, and wily.

Why do we expect obscure scholarly words to be big words, and why do we expect familiar everyday words that “we seem to have known from the time we were born” to be small words? We will begin to answer this bigger “Why” question by looking at some data that will let us ask the following more specific research questions: Do speakers of English share our intuition that terms such as “big word” versus “small word” invoke something about the perceived familiarity of the words? Are obscure “big words” generally bigger than everyday “small words” despite exceptions such as aver as compared to caterpillar? And how do the typical sizes of “big words” and “small words” compare to the typical sizes of English words overall?

What strategies can we use to gather data to answer these questions? One thing we could do is to take a reasonably large collection of words of different sizes and ask a reasonably large number of speakers of English to classify them as being either “big words” or “small words” (with maybe a third intermediate category for “not too big or small”). Another thing we could do is to collect a large dataset of words that someone has already classified as being “big words” or “small words” and then measure their sizes by counting the number of letters in each of the words in the dataset. In this chapter we’ll use data that were collected using both of these research strategies. In the next two sections we’ll introduce the two new types of data that these two strategies yield.

### 2.2. Ordinal categorical data

In Chapter 1, we made a broad distinction between categorical variables and numerical variables. We defined a categorical variable as a set of recorded observations of group or category membership. We pointed out that categorical variables are not
inherently ordered on a number scale. For example, in describing types of letter-to-sound correspondences for the digraph ‘gh,’ we developed a list of words which included the types k, ɡ, and f. The symbols for these sounds might be put into some kind of alphabetic order, but according to which alphabet? The IPA consonant chart lists symbols starting with stops, but the sort function in R puts them in the order that they appear as letters in the current English alphabet. This is based on the order of the Latin alphabet, which originally didn’t include the letter ‘g’ and wrote both ɡ and k with ‘c’. The oldest English alphabet was based on runes, and started with letters for the sounds f, u, θ, o, r, k. So, as you can see, there’s no intrinsic order to these categorical variables.

Similarly, in describing a list of words that you’ve collected to begin to get at the question of why “big words” are called that, you might ask your classmates to categorize them as “big words” (or “high-brow words”) versus “small words” (or “everyday words”). These could be the two types relevant for a variable that we might call “perceived familiarity.” Again, there’s no obvious order here.

Alternatively, if you wanted to get at perceived familiarity more directly, you could use types such as the ones described in Table 2.1. This categorical variable is subtly different from the simple two-way distinction between high-brow words and everyday words because it invokes a rank order or scale. That is, the type “unfamiliar” is understood to be in between the other two types; the words in this category are not as familiar as the words in the first category, but they are not as completely unfamiliar as the words in the last category. When the types of a categorical variable can be rank ordered in this way, it is an **ordinal categorical variable**.

| familiar | “You are familiar with the word and know its meaning.” |
| unfamilar | “You recognize the word but don’t know what it means.” |
| obscure | “You have never seen the word before.” |

The descriptions in Table 2.1 of what it means for a word to be familiar versus more or less unfamiliar are from an experiment that a team of psychologists in David Pisoni’s Speech Research Laboratory did in the early 1980s. They asked Indiana University undergraduates who were native speakers of American English to rate the familiarity of words taken from a couple of popular pocket dictionaries. There were 19,750 words used in the experiment, randomly divided into enough different smaller lists that no participant had to rate too many words, and each word could be rated by 12 of the students who participated in the experiment. Getting 12 different ratings is like the reporter interviewing 12 different observers at an accident. This gives us a database of many words that are in dictionaries that Caronna may have consulted as a college student, each of which has been rated by 12 college students.
ing, then it goes in the “unfamiliar” category. If the majority said they had never seen it before, it goes in the “obscure” category. The left panel in Figure 2.1 shows the distribution of words in this *Hoosier Mental Lexicon* (HML) database across these three familiarity ranks.

We can use these types to check our idea that familiarity is the relevant difference between the “big words” that Caronna chose to put on her list and the “small words” that Lederer chose to use in his essay. That is, for each word on Caronna’s list and for each word in Lederer’s essay, we can check whether it is in the *Hoosier Mental Lexicon* database, and if it is, we can see which familiarity rank it was assigned by the twelve students who judged that word. The middle panel of Figure 2.1 shows the distribution of words on Caronna’s list across the three familiarity ranks and the right panel shows the distribution of words in the paragraphs of Lederer’s essay where he was using only one-syllable words. Note that we had to have an extra “NA” type (for “rating not available”) in the middle panel of the figure, to count the tokens on Caronna’s list that are not in the *Hoosier Mental Lexicon*. Pocket dictionaries can only contain a limited number of words, so the most obscure words that occur in unabridged dictionaries are not included in pocket dictionaries, and thus, are also not in the HML.

Note also that the y-axis scale is different across the three panels. Because the overall number of tokens differs greatly between the *Hoosier Mental Lexicon* and the other two samples, the left panel has a maximum count of over 12,000 tokens, for the number of words in the unfamiliar category, whereas Caronna’s list has just under 200 tokens in that type, and Lederer’s essay has even fewer words in the familiar category, which is the type with the most tokens for that sample. We’ve adjusted the y-axis scales so that these tallest bars are about the same height on the page. This adjustment emphasizes that it’s the relative number of tokens in each category and not the absolute number that is relevant for our question. That is, for example, by making the bar plot of how many of the words on Caronna’s list fall into each of the three types defined by the familiarity ratings in the larger database, we are giving ourselves a very easy way to compare her observations to the observations of the twelve Hoosier students who rated each word in the *Hoosier Mental Lexicon*. By looking at the relative heights of the bars for the same types in the two samples, we can see if Caronna’s “big words” are also unfamiliar or even unknown words.

---

* Pisoni and his colleagues actually used more types than the three described in Table 2.1, to make a seven-category scale ranging from very familiar words down to totally obscure words. However, in picturing the relationship between word length and another variable that they thought might be related to the degree of familiarity, they collapsed the finer-grained scale into just three ordered types based on the average rank that was assigned by the 12 raters. The statements in Table 2.1 are the words that Pisoni and colleagues used to describe the two extreme types and one of the intermediate types at the center of their scale.
Generalizations about the data displayed in Figure 2.1:

1. Although “unfamiliar” words are in the majority, there are more “very familiar” words than there are “totally obscure” words in the Hoosier Mental Lexicon.

2. By contrast, there are no “very familiar” words on Caronna’s list of “My Big Words.”

3. There are proportionally more “totally obscure” words on Caronna’s list of “My Big Words” as compared to the number of such words in the Hoosier Mental Lexicon.

4. There are no “totally obscure” words in the Lederer essay sample.

5. A large majority of words in the Lederer sample are in the “highly familiar” category.

Caronna describes how she started her list of “big words” as follows: “I collected these words when I was in college (early 1990s). When I read or studied for class, I wrote down words I didn’t know.” So if her observations match the observations of the Hoosier college students from a decade earlier, then very few of the words on her list should fall into the “familiar” type. Indeed, as the generalizations in the gray box under Figure 2.1 state, there are no words on Caronna’s list that fall into that “very familiar” category, so the third highest count on Caronna’s list is not the “familiar” category but the “NA” category that we had to add for the 75 words on her list that are not even in the Hoosier Mental Lexicon.

Counting the tokens of different types for a categorical variable gives us a useful way of comparing numbers and of talking about how the types differ in their relative counts. For example, we can see that there are more tokens in the category “totally obscure” than in the category “very familiar” in Caronna’s list of “big” words. This is the opposite of the relationship between these two categories in the Hoosier Mental Lexicon. Also, even though both Caronna’s list and the HML have more “unfamiliar” words than any other type, the HML has fewer “obscure” words in relation to “very familiar” words. Finally, if “small words” are the opposite of “big words” along this dimension of “familiarity” types, then we might predict that the order of counts for the “unfamiliar” and “very familiar” categories from Lederer’s essay would be reversed when
we compare the number of tokens of each type for a dataset of “small words” because more of the shorter words he uses should be “familiar.” And indeed, when we count the tokens of each type in the title and first four paragraphs of Lederer’s essay, we find that this prediction is borne out. Of the 139 different words in that stretch of text, none fall into the “obscure” type, 34 fall into the “unfamiliar” type, and the large majority (the other 105) fall into the “very familiar” type. So we can infer more confidently that the perceived size that is named by the expressions “big word” versus “small word” seems to be related as much to the word’s familiarity as to its actual physical size.

It’s important to remember, though, that it’s only the numbers that we get when we count the tokens that are any kind of inherently ordered quantitative scale here. The categorical variables that we are counting are not themselves inherently quantitative (numerical), and we could, in theory, put the token counts in any order in a table or a figure. In the two panels of Figure 1.2 in Chapter 1, for example, we ordered the bars for the token counts of the types for the categorical variable “letter-to-sound correspondences for the ‘gh’ digraph” according to the relative number of tokens of each type in word-medial or word-final position in the words in the sample that we used to make those counts. This helped us see at a glance that the distribution of types was different in word-initial position. Similarly, we could have ordered the bars in each of the panels of Figure 2.1 by the relative frequency (that is ordered by whichever type had the most tokens) in the Hoosier Mental Lexicon. That would have let us see at a glance that the most common type in this larger wordlist is not the same as the most common type in the Lederer essay. However, it would not have let us see as easily that the two smaller samples differ from the larger HML sample in opposite ways. That is, it would not have let us see as easily that, although the distribution of words on Caronna’s list and the distribution of words from the Hoosier Mental Lexicon peak at the same intermediate “unfamiliar” category, the other words on Caronna’s list are all either in the even less familiar “obscure” category or not even to be found in the dictionaries from which the much larger sample of words in the HML were taken. By contrast, the distribution of words in Lederer’s essay peaks at the “familiar” type and the sample includes no words that are in the “obscure” category.

This idea of an “opposite” direction of difference comes out clearly in Figure 2.1 because of the way that we have exploited the ordinal nature of this categorical variable. That is, we’ve chosen to arrange the counts for the three types in each panel of the figure in a way that puts the “unfamiliar” type in between the “totally obscure” type and the “very familiar” type. This is in keeping with the fact that the three types are ordered along a qualitative scale of familiarity. But that’s an order that comes from our knowledge of the real-world relationships among these categories, and it’s that real-world knowledge that we rely on when we interpret the count of the “NA” category in the middle panel of Figure 2.1.
2.3. Discrete numerical variables

By contrast to a categorical variable, a **numerical variable** is inherently quantitative and ordered by the quantity that it measures. It is a set of recorded observations that are necessarily ordered on the scale of numbers. For example, in discussing the differences between the categories “big words” and “small words” earlier in this chapter, we referred to the numerical variable “number of letters.” This variable is ordered because four letters is always “larger” than three letters, which is in turn “larger” than two letters. That is, the numbers here refer to a scale of countable quantities, where 4 (the count of letters in *aver* and *rive*) is larger than 3 (the count of letters in *the* and *two*), which in turn is larger than 2 (the count in *we* and *to*).

Another distinction that we can make involves what kind of numbers we use in recording numerical variables. Some numerical variables, such as height measurements, have **continuous** values. Depending on where we are in the world, we might measure height in meters and centimeters or we might measure height in feet and inches. What makes these values continuous is that we can always subdivide centimeters or feet into smaller units such as millimeters or inches, or even nanometers, or 1/2 or 1/4 inches, etc. So the number of distinct values for any **continuous numerical variable** such as height measurements is arbitrary, and constrained only by properties of the measuring device. In measuring heights, for example, you might use a ruler or a measuring tape, and it will have marks on it that can only be so thin or so close together.

We’ll talk more about continuous numerical variables in the next chapter and focus here instead on discrete numerical variables. A **discrete numerical variable** is one where the units of measurement inherently cannot be subdivided. For example, if you wanted to compare word lengths in the *Hoosier Mental Lexicon* and in Caronna’s wordlist, you might look at how many letters the spelled form of each word in the list has. Since the letters that we use to write English words can’t be subdivided into fractions of letters, the numbers that you count are always integers that range from 1 letter for words such as *I* and *a* to 19 or more letters for words such as *counterintelligence*. See Table 2.2 for the range of lengths of words in the *Hoosier Mental Lexicon* measured in this way. Here we have ordered the lengths from shortest word spelling to longest word spelling. The token counts for each length ranges from only 1 for the least common length (the 19-letter word *counterintelligence*) to more than 3000 (the 6- and 7-letter words). You can see at a glance that ordering the lengths from the least frequent length to the most frequent length would give a very different order from the order of rows in the table.
Table 2.2. Lengths of spelled words (in number of letters) for the 19,321 words in the *Hoosier Mental Lexicon*, with number of words and examples for each length.

<table>
<thead>
<tr>
<th>NUMBER OF LETTERS</th>
<th>NUMBER OF WORDS THAT LONG</th>
<th>EXAMPLE WORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>a, I</td>
</tr>
<tr>
<td>2</td>
<td>29</td>
<td>ad, is, we</td>
</tr>
<tr>
<td>3</td>
<td>474</td>
<td>ago, ink, moo, oak, she, the</td>
</tr>
<tr>
<td>4</td>
<td>1599</td>
<td>alms, idea, mild, oven, reek, toga</td>
</tr>
<tr>
<td>5</td>
<td>2354</td>
<td>album, inept, mirth, opera, ridge, torso, viola, woozy</td>
</tr>
<tr>
<td>6</td>
<td>3052</td>
<td>aboard, bolero, bounce, indeed, myopia, oyster, ragged, unique</td>
</tr>
<tr>
<td>7</td>
<td>3039</td>
<td>anemone, insular, measles, origami, soldier, tedious, utilize</td>
</tr>
<tr>
<td>8</td>
<td>2616</td>
<td>aardvark, diplomat, indoors, roughage</td>
</tr>
<tr>
<td>9</td>
<td>2316</td>
<td>adventure, davenport, knicknack</td>
</tr>
<tr>
<td>10</td>
<td>1682</td>
<td>alphabetic, derivative, roundhouse</td>
</tr>
<tr>
<td>11</td>
<td>1075</td>
<td>attenuation, dilapidated, forethought</td>
</tr>
<tr>
<td>12</td>
<td>567</td>
<td>appendicitis, recapitulate, malnourished</td>
</tr>
<tr>
<td>13</td>
<td>315</td>
<td>autobiography, kaleidoscopic, syllabication</td>
</tr>
<tr>
<td>14</td>
<td>106</td>
<td>aggrandizement, whippersnapper</td>
</tr>
<tr>
<td>15</td>
<td>50</td>
<td>Americanization, postconsonantal</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>arteriosclerosis, multimillionaire</td>
</tr>
<tr>
<td>17</td>
<td>8</td>
<td>indestructibility, ultraconservative</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>antivivisectionist</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>counterintelligence</td>
</tr>
</tbody>
</table>

2.4. Picturing numerical data

Recall how we used a bar plot to show the distribution of the frequency of different types of categorical data, like the graphs in Chapter 1, or in Figure 2.1 in this chapter. *(Frequency, here, means how many tokens there are of a particular type, or how frequently that type is represented. **Relative frequency**, then means how frequent one type is relative to the other types.)* We can use a bar plot to look at the distribution of values for discrete numerical variables, too, but in this case, we take full advantage of the natural quantitative order that the data type gives us when we decide how to arrange the bars. Specifically, we order the bars from the bar for the smallest numerical value to the bar for the largest numerical value, and we have as many bars as there are numbers in the range of values that we show. This means leaving a gap for an “invisible” bar at any value where there are no tokens.

*Figure 2.2.* Histogram of word lengths (in number of letters) in the *Hoosier Mental Lexicon*.

*Figure 2.3.* Histogram of word lengths in Carol Caronna’s collection of “big words.”
Figure 2.4. Histogram of word lengths in the title and first four paragraphs of Richard Lederer’s essay on “The Case for Short Words” with each different word counted only once.

Thus, Figures 2.3 and 2.4 show histograms of the same numerical variable “number of letters in the spelled word forms” in Caronna’s list of “big words” and in all of the different “short words” that Lederer used in the title and first four paragraphs of his essay. Notice that the y-axes in these bar plots are adjusted to the size of the sample. Here, the range of values shown on the y-axis for the HML is much larger (from 0 to about 3000) than those for Caronna’s list of “big” words (from 0 to about 80) and the words in the excerpt of Lederer’s essay (from 0 to about 50). This is because there are so many more words in the HML than in Caronna’s list or Lederer’s essay.

Also, although the y-axes of Figures 2.3 and 2.4 differ from that of Figure 2.2, we’ve kept the x-axes from Figure 2.2, even though that means we have lots of “invisible” bars for word lengths that are attested in the HML but not in one or the other of the shorter lists. Keeping the x-axis range constant in this way lets us compare very easily the distribution of lengths across the three samples. You can see at a glance that each of the three bar plots shows a very prominent peak at a single value, but it is at different places along the scale of numbers on the x-axis. The peak is at 6 letters in the bar plot for words in Hoosier Mental Lexicon, at 9 letters in the bar plot for Caronna’s “Big Words”, and at 4 letters in the bar plot for the distinct “small words” that Lederer used. Having a single peak makes it easy for us to see that the most common length for the “big words” is indeed longer than the most common length for the words in the Hoosier Mental Lexicon, even though the longest of the “big words” is six letters.

Figure 2.2 illustrates this type of bar plot for the numbers in Table 2.2. It shows the distribution of lengths of word spellings in the Hoosier Mental Lexicon, and the different lengths are ordered on the x-axis from the shortest words, such as a and I, to the longest word, which is counterintelligence. That is, the “categories” plotted on the x-axis here are the inherently ordered possible values, arranged from the length of the words with the fewest letters (just 1) to the length of the word with the most letters. The y-axis represents the token counts, or frequencies, of each of these numerical values. For example, in the HML (Figure 2.2), there are about 500 tokens of 3-letter words. That is, the “category” of 3-letter words has a frequency of around 500, meaning that there are about 500 occurrences of this category. A bar plot that shows the distribution of values for a numerical variable in order, with an “invisible” bar (having a height of “0” tokens) specified at any value where no tokens are observed, is a kind of a histogram.
shorter than the longest word in the larger sample. Plotting
the data as histograms also makes it easy to see that every one
of Lederer’s “small words” is shorter than the most common
length for the “big words” sample.

2.5. Answering our research question

The questions that we asked at the beginning of this chapter
are: Are “big words” less familiar than most words that people
know and are “small words” more familiar”? Conversely, are
“big words” really bigger than “small words,” and how do the
sizes of “big words” and “small words” compare to the size of
words in English overall? In section 2.2, we used a large sam-
ple of words that had been rated for familiarity along a scale
of types of an ordinal categorical variable ranging from “to-
tally obscure” to “very familiar,” and saw where all of the
words in a sample of “big words” and all of the words in a sam-
ple of “small words” fall along this familiarity rating scale. We
found that many of the “big words” were ones that the Hoo-
sier undergraduate raters had judged to be in the “obscure”
category, whereas the majority of the “small words” were ones
that the raters had judged to be in the category at the other
end of the scale.

In section 2.4, we then compared the number of letters in the
words on Caronna’s list of “big words” with the number of let-
ters in all of the different “small words” that Lederer chose to
use in the title and first four paragraphs of his essay. We saw
that “big words” indeed tend to have more letters than “small
words.” The tallest bar in the histogram for the “big words” is
the one for 9-letter words, whereas the tallest bar in the bar
plot for “small words” is the one for 4-letter words. That is,
nine letters is the most common size of our “big words” sam-
ple, whereas four letters is the most common size for our
“small words” sample. We also compared the number of let-
ters in “small words” and “big words” with the number of let-
ters of words in the Hoosier Mental Lexicon. If we take this
wordlist as a fairly representative sample of the size of words
in the English language as a whole, then we can see that “big
words” are, in fact, bigger than average, and “small words”
are, in fact, smaller than average. The tallest bar in the histo-
gram showing word lengths in the Hoosier Mental Lexicon is
the one at 6 letters, closely followed by the bar for 7 letters.

2.6. Summary

When we talk about data, we are talking about sets of reliably
recorded observations that we can use to reason numerically
about the world. Some records are inherently quantitative, de-
signed by counting or measuring something. For example, we
can measure word lengths by counting the number of letters
or the number of sounds. Other records are qualitative judg-
ments about category membership. For example, we can
group words into categories such as “familiar words” versus
“obscure” ones. We can derive useful quantitative measures of
such categorical variables by counting the number of tokens
for each type. If the categories have some kind of order to
them, as in a familiarity rating scale, we can arrange the token
counts according to that order, to see the distribution of tokens along the qualitative scale of types. Plotting the numbers that we get in bar plots and histograms is a useful way to look for generalizations about the population.

### Summary of key terms

**Ordinal categorical variable**: A set of recorded observations of category membership where the category types might be conceptualized as ordered along some kind of qualitative scale. For example, the category types: “totally obscure,” “unfamiliar,” and “very familiar.”

**Numerical variable**: A set of recorded observations that are inherently ordered on a scale of numbers. Example: height (for people) or number of letters (for spelled forms of words).

**Continuous values**: Values that you can always subdivide into smaller units. The number of distinct values for a continuous variable is arbitrary. Example: height. If your range is 70-170 cm, you may have arbitrarily closely spaced fractions or decimals, such as the possible values {72.5, 72.49, 72.486458, 72 ¼ ...}.

**Discrete values**: Values that are always integers and can’t be subdivided into fractions. For a definite range of values, you will have a specific number of possible values. Example: number of letters. If your range is 1-15 letters, you can only have 15 possible values {1,2,3,4...15}.

**Histogram**: A special type of bar plot used to represent the frequencies for different values of a discrete numerical variable. Specifically, the bars of a histogram must be arranged in strict numerical order and there must be a space left for an “invisible” bar for any value where there are no tokens within the range of values for the sample. Example: The plots in Figures 2.2-2.4. (In the next chapter, we’ll show you what needs to be added to this definition in order to define what a histogram is for a continuous numerical variable.)

### Section 2.7 R code
Part 1. Making bar plots to compare distributions of three samples of a categorical variable

First, use the \texttt{c()} function to create a vector (a simple series of data, where all of the data are of the same type) to tell R how many words in the \textit{Hoosier Mental Lexicon} fall into each of the three familiarity types defined in Table 2.1. We will call this vector \texttt{example1}. Simply give the token counts as arguments for the function, putting them in the same order that you want to display them in the bar plot, and separating them by commas. The operator = to the left of the \texttt{c()} is the “assignment” operator, which assigns this vector to a variable named \texttt{example1}.

\begin{verbatim}
example1 = c(1880, 12632, 4809)
\end{verbatim}

After making this assignment, you can refer to the vector of numbers using the name \texttt{example1}. For example, typing the name all by itself makes the three numbers show up on your R console window. That is, typing this:

\begin{verbatim}
example1
\end{verbatim}

results in this:

\begin{verbatim}
[1] 1880 12632 4809
\end{verbatim}

Since these three values are all numbers, you can use the vector as an argument for functions such as the \texttt{sum()} function, which adds up the elements that are given to it as arguments, just as we have done in the preface and first chapter. Try typ-
ing each of the following commands, to see how the name of the vector can stand in for the vector itself.

\[
1880 + 12632 + 4809
\]
\[
\text{sum(c(1880, 12632, 4809))}
\]
\[
\text{sum(example1)}
\]

You can use this to check to make sure you typed the numbers correctly when you assigned the vector to the `example1` variable. That is, all three of the above commands should return 19321, which is the total number of words in the *Hoosier Mental Lexicon*.

Next, assign a name to each of the three numbers in this vector, identifying what the category is. Use the same order as you did in the command above. The `names()` function takes a named vector, such as the vector `example1`, as its argument, and it either returns the vector of names for the elements of the vector or (as here) lets you assign names to the elements of the vector.

\[
\text{names(example1) = c("obsc", "unfam", "fam")}
\]

Notice that the three values in the vector of names are each set off with quotation marks. This is because these values are specifically “character strings” — a different type of R data from numbers. You can appreciate the difference if you try to sum the values in a vector of character strings that is superficially similar to the `example1` vector, as shown below.

\[
\text{sum(c("1880","12632","4809"))}
\]

This gives us an error message that tells us that `sum()` is an invalid operation for character strings.

Next, use the `barplot()` function to make a bar plot, where the heights of the bars are the numbers in the `example1` vector. This function has one obligatory argument, a vector of numbers to use for the heights, as shown below.

\[
\text{barplot(example1)}
\]

Don’t forget the x-axis and y-axis labels!

\[
\text{barplot(example1, xlab="familiarity types", ylab="number of words")}
\]

You can also add a main title at the top, and once you have made a bar plot, you can draw a box around it using the `box()` function.

\[
\text{barplot(example1, xlab="familiarity types", ylab="number of words", main="types of words in the HML")}
\]

\[
\text{box()}
\]

You can change the range along the y-axis, so that the top of the highest bar doesn’t hit the box you’ve drawn, using the `ylim` argument in the `barplot()` function.

\[
\text{barplot(example1, ylim=c(0,13150), xlab="familiarity types", ylab="number of words", main="types of words in the HML")}
\]

\[
\text{box()}
\]
You can save the graph using the “Save as ...” command in the “File” drop-down menu at the top of the screen.

Now, you can now do the same three steps for Caronna’s list to make another bar plot.

    example2 = c(75,84,198,0)
    names(example2) = c("NA", "obsc", "unfam", "fam")
    barplot(example2, ylim=c(0,220),
            xlab="familiarity types", ylab="number of words", main="types of words in Caronna’s list")

    box()

Then, let’s read in the file Ch02.Textfile1.txt in order to find the counts of familiarity types from Lederer’s small words to compare. This is different from the procedure for the first two graphs, because instead of typing the numbers in as a vector, we will read in a whole file and let R make a table, creating tallies of one column to get our counts.

First you will need to download the file Ch02.Textfile1.txt from the course website at http://hdl.handle.net/1811/77848, or directly at http://kb.osu.edu/dspace/bitstream/handle/1811/77848/Cho2.Textfile1.txt. This file is a plain ASCII text file that has as many rows as there are words, plus one extra row at the top which is a “header” row identifying the sequence of variables that are recorded on each row.

Remember to set your directory to wherever the files are located. You can do this using the “Change Working Directory ...” command, which is under the “Misc” pull-down menu on a Mac, or the “Change Dir...” command under the “File” pull-down menu on a PC running Windows. Make sure your console window is active (you can click in it to be sure), so that the appropriate choices are available from the pull-down menu.

Then use read.delim() to read the file into R:

    lederer = read.delim("Ch02.Textfile1.txt",
                        header=TRUE)

Use the head() function to see what is in it:

    head(lederer)

    word freq familiarity famType
    1   and   18      7.0000 very fam.
    2   the   18      7.0000 very fam.
    3 words  14      7.0000 very fam.
    4  like    9      7.0000 very fam.
    5   you    9      7.0000 very fam.
    6  that    8      6.4167    unfam.
The object that you read in this way is called a data frame, which is a two-dimensional table of information organized in rows and columns. Each column is a vector of elements, with as many elements as there are rows of data in the data file. That is, each element corresponds to one of the pieces of information recorded for each row of data in the data file that you read in. Since there are four pieces of information in each row of the Ch02.Textfile1.txt data file, there should be four columns in the data frame, and since there were 139 words in Lederer’s list, there should be 139 rows in the lederer data frame. You can use the `dim()` function to make sure that the dimensions of the data frame are indeed 139 rows (for the 139 words) in 4 columns (for the four kinds of information about each word). The `dim()` function will always list the number of rows first, then columns.

```
dim(lederer)
```

The above command should return something that looks like the following:

```
[1] 139   4
```

As you can see, there are indeed four columns. The `head()` command above showed that these columns correspond to each piece of information that is included about each word. The first column gives the word itself. The second column contains a frequency count, which tells how often the word occurs in the essay. The third column contains the average familiarity rating as found in the *Hoosier Mental Lexicon*. And the fourth column contains the familiarity type, as explained in Table 2.1.

In order to make our bar graph, we want to access the 4th column with the familiarity types. Recall that we can use square brackets to refer to values from specific rows and/or columns in a dataframe, so that if we want to isolate the 4th column, we could refer to it in this way:

```
lederer[,4]
```

Which gives us a vector of only the familiarity types found in that column.

But there is another way specifically of referring to entire columns of data in a dataframe with a header, and that is to use the variable name (in this case, lederer) plus a dollar sign ($) which tells R that we are only interested in one specific column of that dataframe, plus the column name that is specified in the header (in this case, famType). Together, it looks like this:

```
lederer$famType
```

and gives us the same output as we got from typing in `lederer[,4]`, which might look like this at the top:

```
[1] very fam. very fam. very fam. very fam. very fam. very fam. unfam.
[7] very fam. unfam. very fam. unfam. very fam. unfam.
```
and like this at the bottom:

\[
\begin{array}{c}
133 & \text{unfam. very fam. very fam. very fam. very fam. very fam.}
\end{array}
\]

\[
\begin{array}{c}
139 & \text{unfam.}
\end{array}
\]

Levels: unfam. very fam.

(You will get different numbers at the beginning of each row if your R console window is wider or smaller than the one that we had when we typed this command. Don’t worry about that difference. It’s not going to affect anything that you do with this vector.) The last row in the display – i.e., the one that says, Levels: unfam. very fam. – tells you that the type of information in this 4th column is a categorical variable with these two types. That is, R uses the term “levels” to refer to the types of a categorical variable, and you can retrieve just the list of types using the \texttt{levels()} function, like this:

\[
\text{levels(lederer$famType)}
\]

Another very useful function in R is the \texttt{table()} function. This instructs R to look for all the possible values in a vector and tally up how many times each type occurs. So, in the 4th column of \texttt{lederer}, R will find instances of "very fam." and "unfam.", and count up how many there are of each.

The \texttt{table()} function requires the name of the vector as its main argument, and we can use the \texttt{lederer$famType} shortcut we just learned to supply this information, to create a table of token counts of the different types of familiarity:

\[
\text{table(lederer$famType)}
\]

Which will give us this output:

\[
\begin{array}{c}
\text{unfam. very fam.}
\end{array}
\]

\[
\begin{array}{c}
34 & 105
\end{array}
\]

Note, however, that there are no “obscure” words in this file. So in order to be able to compare the distribution of familiarity types in this list to the distributions in the HML and in Caronna’s list, we will need to add that as a possible type. We can do this very simply by creating a vector that includes the table of counts we created and a zero for the obscure words count.

\[
\text{example3} = \text{c}(0,\text{table(lederer$famType)})
\]

We will then have to adjust the names to match those in Figure 2.1, so they will line up with the three values we now have:

\[
\text{names(example3) = c("obsc","unfam.","very fam.","")}
\]

Then we can make a bar plot:

\[
\text{barplot(example3, ylim=c(0,120), xlab=}
\]

"familiarity types", ylab="number of words", main="types of words in Lederer's essay")

\[
\text{box()}
\]
Part 2. Making histograms to compare distributions of samples of a numerical variable.

For these plots, the vectors need 19 values, for each of the integers from 1 through 19, the range that we have specified in Figures 2.2 through 2.4. We have to go up to 19 because that is the length of the longest word in the HML, even though it is difficult to see the very short bars in Figure 2.2. If you look at the numbers in Table 2.2, you'll see that there are two tokens of 18 letter words and one token that is 19 letters long. Because we want to compare the distributions of Caronna’s “big words” and Lederer’s “small words” to the overall distribution in the HML, we should have the same values on the x-axis for all three histograms. And since it’s less practical to type in 19 token counts by hand, let’s take advantage of some other R functions, to read in the word lists, calculate the length of each spelled word form, and make R count the tokens for each length for us.

Before you can start, you need to download the files Ch02.Textfile2.txt (the HML wordlist) at http://kb.osu.edu/dspace/bitstream/handle/1811/77848/Ch02.Textfile2.txt and Ch02.Textfile3.txt (Caronna’s word list) at http://kb.osu.edu/dspace/bitstream/handle/1811/77848/Ch02.Textfile3.txt, and you will need the Ch02.Textfile1.txt (which you will have already used in Part 1), from the textbook web site, and set your directory to wherever the files are located. Each of these files is a plain ASCII text file that has as many rows as there are words, plus one extra row at the top which is a “header” row identifying the sequence of variables that are recorded on each row. For example, the first 13 rows of the Ch02.Textfile3.txt file look like this:

"word"  "familiarity"  "famType"
"axiomatic" 2.8333  "obscure"
"abeyance" 3.5  "unfamiliar"
"attenuate" 4.5833  "unfamiliar"
"anodyne"3.1667  "unfamiliar"
"apposite" 4  "unfamiliar"
"assiduous" 4.75  "unfamiliar"
"attrition" 4.8333  "unfamiliar"
"abrogate" 3.0833  "unfamiliar"
"ameliorate" 3.1667  "unfamiliar"
"assuage"3.0909  "unfamiliar"
"ad hoc" NA  "out"
"arcane" 2.6667  "obscure"

On each row after the first, you can see one of Caronna’s “big words” in its ordinary spelled form, followed by a number that is the average familiarity rating for the word, if the word is also in the Hoosier Mental Lexicon (with “NA” as the value if it’s not), and the type to which we assigned the word. The fa-
miliarity numbers represent the conversion of an ordinal categorical variable (with types such as “obscure”, “very familiar,” etc.) to a numerical variable, along a scale of 1 to 7, corresponding to the seven original familiarity ratings used in the study.

After you have set the working directory to where you put the files, use the `read.delim()` function and the `=` operator to read in the `Ch02.Textfile3.txt` file and assign it to the variable `big`.

```r
big = read.delim("Ch02.Textfile3.txt")
```

You can use the `dim()` function check to make sure that the dimensions of the data frame are indeed 357 rows (for the 357 words) in 3 columns (for the three types of information about each word).

```r
dim(big)
```

The above command should return something that looks like the following:

```
[1] 357  3
```

You can check to make sure that the three columns are called the same thing that they’re called in the header row in the file, using the same `names()` function that you used in Part 1 to give names to the elements in the two vectors you made, like this:

```r
names(big)
```

You can refer to each column in a data frame by its name. For example, to check what’s in the third column, you can type the following command, which then prints out to the screen all 357 elements in the “famType” column:

```r
big$famType
```

You can use the `table()` command to count how many tokens there are in each of these types, like this:

```r
table(big$famType)
```

This should return a named vector of numbers like the one you made in Part 1, but in this case, “out” represents the NA values which were not even listed in the HML, and the “familiar” type is not listed because there were zero tokens of that type:

```
obscure   out    unfamiliar
     84      75     198
```

Of course, what you want to do in this part is to get a named vector of counts for a different bit of information that isn’t yet available — namely, the numerical variable of word lengths...
that we plotted in the histogram in Figure 2.3. You can use the `nchar()` function to obtain the numbers you’ll need in order to create this variable. The `nchar()` function counts the number of characters in a character string. You can test this by typing any of the following commands:

```
  nchar("axiomatic")
  nchar("abeyance")
  nchar("word")
  nchar("familiarity")
  nchar("famType")
  nchar(names(big))
```

If you are following along in R, you can see how embedding the `names(big)` function inside of the `nchar()` function gave us a vector of three numbers corresponding to the number of letters in the names of the columns for the `big` data frame. So we ought to be able to take the vector of words in the first column of the data frame (by using `big$word`), and use `nchar()` to get a vector of numbers which gives the number of letters in each word.

However, when R read the data into the big data frame, it treated the first column as another categorical variable, just like the third column. So, to count the number of letters in each word in the `big$word` column, we need to tell R to treat the words as raw character strings, instead of as the names of types for a categorical variable. We can use the `as.character()` function to do this, as shown below:

```
  nchar(as.character(big$word))
```

This returns a vector of the number of characters in each word, from the first word (`axiomatic`, with 9 characters) to the 357th word (`zeitgeist`, also 9 characters long).

We can embed that command further inside the `table()` function to make a vector of token counts for each number of characters. That is, R will look at the list of counts created by using the `nchar()` function, and tally up how many instances there are of each number.

```
  table(nchar(as.character(big$word)))
```

But in order to make a graph of these counts, it will be useful to assign this vector a variable name, such as `BigLetters`, so that we can refer back to it more easily:

```
  BigLetters = table(nchar(as.character(big$word)))
```

Then, if you type the variable name `BigLetters` all by itself, it should return the following named vector:

```
  4  5  6  7  8  9 10 11 12 13
  4 17 40 53 72 75 50 25 15  6
```

Note that here, the first row of “numbers” you see is actually not a vector of numbers, but a vector of character strings that
are the names of the elements in the second row of numbers, as you can see if you type:

names(BigLetters)

You can now use BigLetters as the argument for the barplot() command, to make your histogram, like this:

barplot(BigLetters)

However, note that this doesn’t give you the range that you want, if you want to recreate Figure 2.3 with the same x-axis range. To do that, you can fill in the gaps for the numbers 1 through 3 at the beginning and the numbers 14 through 19 at the end, creating a vector of the existing values plus the missing values like this:

AllBigLetters = c(0,0,0,BigLetters, 0,0,0,0,0)

Note that there are no names for the three counts of 0 at the beginning and the six counts of 0 at the end. You could assign these names one by one, like this:

names(AllBigLetters)[1] = "1"
names(AllBigLetters)[2] = "2"
names(AllBigLetters)[3] = "3"
names(AllBigLetters)[14] = "14"
names(AllBigLetters)[15] = "15"
names(AllBigLetters)[16] = "16"

However, this is tedious. Here is a more convenient way to assign a sequence of 19 character strings which are the numerals 1 through 19 to the names values of the vector all at once, so that we have the names for all 19 types:

names(AllBigLetters) = as.character(1:19)

The “:” operator here means return the sequence of integers that starts with the number before the operator and ends with the number after the operator. And the as.character() function tells R to convert the sequence of numbers into a sequence of character strings. Now we can use this new, complete vector to fill in the argument for the barplot() command to make the histogram:

barplot(AllBigLetters)

We might want to change the y-axis limits (ylim) argument to make the bars fit better inside the box. Since the tallest bar has 75 tokens, we can increase the upper end to 80 so that the tallest bar (the bar for 9-letter words) doesn’t run into the box we want to draw around the graph. We also need to fill in labels for the x- and y-axes and a main title:

barplot(AllBigLetters,ylim=c(0,80), xlab="number of letters in Caronna's big
You will probably want to resize the graph so that all of the types show along the x-axis. This can be done by clicking the lower right corner of the graph and dragging it until you get the size and shape you want. Once you have saved the graph, any information that is not visible is not saved in the pdf file, so you need resize it before saving.

In summary, here again are the six steps to recreate the histogram in Figure 2.3.

```r
big = read.delim("Ch02.Textfile3.txt", header=TRUE)
head(big)
BigLetters = table(nchar(as.character(big$word)))
AllBigLetters = c(0,0,0,BigLetters, 0,0,0,0,0)
names(AllBigLetters) = as.character(1:19)
barplot(AllBigLetters,ylim=c(0,80),
   xlab="number of letters in Caronna’s big words",ylab="number of words of that length",main="Caronna’s big words")
```

Now do the same thing for the `Ch02.Textfile2.txt` file, which contains the data from the Hoosier Mental Lexicon.

```r
hml = read.delim("Ch02.Textfile2.txt", header=TRUE)
dim(hml)
names(hml)
The dim() command should return the following information:
[1] 19321     6
And the names() commands should return the following information:
[1] "orthography" "HMLbet"      "frequency"
   "familiarity" "densityA"
[6] "densityB"
It’s the orthography column that contains the words, so we want to count the number of characters in each word in the vector hml$orthography.

```r
nchar(as.character(hml$orthography))
```

We let R do the work of tallying up the token counts for each type, assigning it to a variable name so we can refer back to it:
Since there are tokens at all 19 lengths in the data frame for Hoosier Mental Lexicon, we can use this variable name directly inside the `barplot()` function.

```r
barplot(hmlLetters)
```

Include labels and change the y limits so that no bars touch the box.

```r
barplot(hmlLetters, ylim=c(0,3100),
        xlab="number of letters in words in the HML", ylab="number of words of that length", main="Words in the HML")
```

In summary, because we don’t need to change anything in the vector of counts, you can recreate the histogram in Figure 2.2 using fewer steps:

```r
hml = read.delim("Ch02.Textfile2.txt", header=TRUE)
```

```r
head(hml)
```

```r
barplot(table(nchar(as.character(hml$orthography))), ylim=c(0,3100),
        xlab="number of letters in words in the HML", ylab="number of words of that length", main="Words in the HML")
```

Notice that the bars for the last few values look like they’re zeros, just as in the histogram you made for the big data frame, but typing the following command all by itself tells you that there are word tokens for these values with very short, nearly “invisible” bars:

```r
table(nchar(as.character(hml$orthography)))
```

### Adapting the R code to recreate Figure 2.4:

Try to fill in the appropriate code for each step before looking to the end for the solution

1. Read in the file "Ch02.Textfile1.txt" using the `read.delim` function

```r
hml = read.delim("Ch02.Textfile1.txt", header=TRUE)
```

2. Find out what the first few lines contain, using the `head()` function

```r
head(hml)
```

3. Create a table of the character counts of the word column in the Lederer file, and assign a variable name (hint, break this up into grouped parentheticals, like this

```r
variablename = table(nchar(as.character( hml$orthography))), ylim=c(0,3100),
        xlab="number of letters in words in the HML", ylab="number of words of that length", main="Words in the HML")
```

where you will replace the `variablename`, `filename`, and `columnname` with the appropriate names.
4. Add the appropriate number of zeros to fill in the missing types, so that you have the same number of types (19) as we have been using in Figures 2.2 and 2.3.

5. Assign the names 1 through 19 to the table to all the types, as you did above, using `as.character(1:19)`.

6. Use the `barplot()` function to create this type of histogram, giving it appropriate x and y labels, and a main title.

Solution:

```r
lederer = read.delim("Ch02.Textfile1.txt", header=TRUE)

head(lederer)

SmallLetters = table(nchar(as.character(lederer$word)))

AllSmallLetters = c(letters,0,0,0,0,0,0,0,0,0)

names(AllSmallLetters) = as.character(1:19)

barplot(AllSmallLetters, xlab="number of letters in Lederer’s small words", ylab="number of words of that length", main="Lederer’s small words")
```

2.8. Exercises

1. For each of the following populations, say whether it is an ordinal categorical variable, a discrete numerical variable, or a continuous numeric variable. Briefly explain how you can tell, as shown in the following example:

   Height (in cm) of each of the players in the FIFA men’s division: *continuous numerical, because the values are inherently ordered (150 is inherently smaller than 174 which is inherently smaller than 180) and centimeters can be divided into smaller units, such as millimeters*

   a. Height (classified as “tall”, “average”, or “short” by the team’s manager) of each player on each team in the FIFA women’s division.

   b. Number of pages in each book in a collection of textbooks written in English.

   c. Length of each book (in minutes) of recordings a collection of audiobooks.

   d. Length of first sentence (in number of words) in each the collections of books.

   e. Grade level of each book, as rated by an experienced public school teacher.

   f. Suitability of each book for use in your local school system, judged on a 3-point scale from “not suitable” to “very suitable”.

78
2. For each of the following research questions, explain what the population is, and how you would find a representative sample.

a. How tall are the undergraduate students at your university?

b. On average, how many syllables does an English word have?

c. Are typical sentences in the *Columbus Dispatch* newspaper longer or shorter than those in the Cleveland *Plain Dealer*?

d. Are typical sentences in the *Columbus Dispatch* newspaper more difficult or less difficult than those in the Cleveland *Plain Dealer*?

e. Did President G. W. Bush make more or fewer pauses in his State of the Union addresses than President Obama?

3. Read each of the following descriptions of data samples and then say:

a. what the research question is

b. what the variable is

c. whether it is a categorical variable or a numerical variable

d. if it is a categorical variable, whether the data can be treated as ordinal categorical data

**Sample 1.** Sue has made a list of 10 common words that she finds difficult to spell. The list includes words such as *immediately, separately*, and *definitely*. She would like to find out how difficult these words are to spell for people her age. To do this, she says each word on the list to 34 classmates in one of her college courses, and they write it down. She looks at the 34 spelled forms for each word and finds, for example, that 19 of her classmates spelled *definitely* correctly as *definitely*, 8 spelled it as *definately*, 6 spelled it as *definetely* and 1 spelled it as *defnitely*. For each word, she counts the number of spelled forms that were correct and the number that were incorrect, to see if some of the words are harder than others.

**Sample 2.** Sue finds that 2 of her classmates spelled all 10 words correctly, 5 made an error on only 1 word, 7 made errors on 2 of the words, 11 made errors on 3 of the words, 6 made errors on 4 of the words, 2 made errors on 5 words, and 1 made an error on 6 words. She repeats the experiment with the 23 people in the Aikido class that she teaches in a local retirement community to see whether older people make fewer mistakes.

**Sample 3.** Sue shows her results to her classmates, and Joe points out that some of the spellings seem less incorrect than others. For example, he thinks *defiantly* is al-
most correct by comparison to *definitely* and *defanitely*. Several of their other classmates nod in agreement. Sue wonders whether the older people in her Aikido class also share Joe’s intuitions. She makes a list of the different incorrect spellings for each word and adds a list of judgments after each incorrect spelling so that each row on the list looks like this:

*definitely* spelled “defiantly”: a) almost correct  
b) incorrect  
c) extremely bad mistake

She gives the questionnaire to her 34 classmates and also to the 23 people in her Aikido class, asking them to circle one judgment for each form. For each spelled form, she counts the number of questionnaires for each of the two age groups where “a) almost correct” is circled, the number of questionnaires where “b) incorrect” is circled, and the number of questions where “c) extremely bad mistake” is circled.

4. a. Use the R code given in part 1 of section 2.8 to recreate the three bar plots in Figure 2.1.  
b. Use the R code given in part 2 of section 2.8 to recreate the histograms in Figures 2.2, 2.3, and 2.4.

5. The `subset()` function is used to extract subsets of rows that fit certain properties from a data frame. For example, the following command would extract the 4809 rows in the `hml` data frame for the subset of words that have a familiarity rating of 7:

```r
subset(hml, familiarity==7)
```

Similarly, the following command would extract the 1880 rows in the `hml` data frame for the subset of words that have a familiarity rating less than 3, and assign it to a new data frame with the variable name `obscure`:

```r
obscure = subset(hml, familiarity<3)
```

a. Use the `subset()` function as shown above to make a histogram of word lengths for just the “obscure” words in the *Hoosier Mental Lexicon*.  
b. Compare the distribution of values to the distributions in Figures 2.2, 2.3, and 2.4. Where are most of the values clustered? Where is the peak? Which histogram does this distribution of obscure words from the HML most resemble?

6. The file `Ch02.Textfile4.txt` contains values for six variables for each of the words in the Moe, Hopkins, and Rush (1982) word list that was introduced in Chapter 1. The variables are (1) the orthographic form of each word, (2) its pronunciation using an ASCII transcription, (3) its frequency (i.e., the total number of times the children used the word in their conversations with the interviewers), (4) the number of letters in the spelled form, (5) the number of vowel and consonant sounds in the pronunciation, and (6) the number of syllables. Reading the file into a data frame and examining the first 6 rows looks like this:
moe=read.delim("Ch02.Textfile4.txt")

head(moe)

<table>
<thead>
<tr>
<th>orth</th>
<th>pron</th>
<th>freq</th>
<th>noLetters</th>
<th>noPhons</th>
<th>noSyls</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>x0</td>
<td>6716</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>AARDVARK</td>
<td>a1rdva2rk</td>
<td>6</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>ABCS</td>
<td>e1bi1si1z</td>
<td>7</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>ABLE</td>
<td>e1bx0l</td>
<td>12</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>ABOARD</td>
<td>x0bO1rd</td>
<td>2</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>ABOMINABLE</td>
<td>x0ba1mx0nx0bx0l</td>
<td>8</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

a. Using the code in Part 2 of Section 2.9 as a model, write your own code to create a histogram like the one in Figure 2.2 for the number of letters in each word in the Moe et al. word list, using the noLetters column.

b. Compare the distribution of lengths for these words with the distribution of lengths for the words in the three datasets plotted in Figures 2.2 through 2.4. Where are most of the values clustered? Where is the peak? Which histogram does this distribution of children’s words most resemble?

c. Now make another histogram for the number of sounds in each word in the Moe, Hopkins, and Rush word list, using the noPhons column.

d. Compare the distribution of values for the two ways of counting a word’s length (i.e., letters versus sounds). Are the distributions the same? Where is the peak in each? If one way of calculating length gives you smaller values, why is there this difference? (Hint: Think of discrepancies such as the number of letters used to spell the k sound at the beginning of the word kick versus the end of the word kick.)

7. Download the file Ch02.Textfile5.txt from the course website. Contained in this file are the words to a famous German poem by Friedrich Schiller: “An die Freude”, known in English as “Ode to Joy”. Use the R code provided to help you do the following:

a. Read in the file and create a histogram of the word lengths of each unique word in the poem.

freude=read.table("Ch02.Textfile5.txt", as.is=TRUE)

uniquefreudeLetters=table(nchar(as.character(names(table(freude)))))

hist(uniquefreudeLetters, xlab="number of letters in words of a poem", ylab="number of different words", main="An die Freude")

# This is what the counts are, btw.
2 3 4 5 6 7 8 9 10 11 12 13 14
9 30 46 43 64 32 26 13 10 6 4 1 2
b. Now create a histogram of the word lengths of all of the words in the poem (counting each occurrence of a word separately when a word is used more than once).

```r
freude=read.table("Ch02.Textfile5.txt", as.is=TRUE)
freudeLetters=table(nchar(as.character(freude[,1])))
hist(freudeLetters, xlab="number of letters in words of a poem", ylab="number of words (counting occurrences separately)", main="An die Freude")
```

# This is what the counts are, btw.
2 3 4 5 6 7 8 9 10 11 12 13 14
24 108 65 51 82 40 28 15 10 10 4 1

c. Describe the two distributions, including similarities and differences. Why do you suppose the shape of the distribution changes when letter counts of all occurrences of a word are counted separately? (Hint: think about some of the most frequently repeated words in English, such as a, an, the, and that)

8. Table 2.3 shows words from Caronna’s word list that were listed as NA, that is, not appearing in the Hoosier Mental Lexicon. Count the number of letters in each word and create a histogram of word lengths by character count, as in the exercises above. Compare the distribution of values to the distributions in Figures 2.2, 2.3, and 2.4. Where are most of the values clustered? Where is the peak? Which histogram does this distribution of obscure words most resemble?

**Table 2.3.** Words on Caronna’s list of “My Big Words” which are not among the words in the Hoosier Mental Lexicon.

<table>
<thead>
<tr>
<th>Word</th>
<th>Word</th>
<th>Word</th>
<th>Word</th>
<th>Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>abscond</td>
<td>ad hoc</td>
<td>ad hominem</td>
<td>apocryphal</td>
<td>bête noire</td>
</tr>
<tr>
<td>chiliasm</td>
<td>chimera</td>
<td>clement</td>
<td>cognoscente</td>
<td>concatenate</td>
</tr>
<tr>
<td>concomitant</td>
<td>compendious</td>
<td>comportment</td>
<td>dalliance</td>
<td>de rigueur</td>
</tr>
<tr>
<td>demagogue</td>
<td>diaspora</td>
<td>dithyramb</td>
<td>dyspeptic</td>
<td>effete</td>
</tr>
<tr>
<td>efficacy</td>
<td>effluence</td>
<td>effluvium</td>
<td>ethology</td>
<td>exogenous</td>
</tr>
<tr>
<td>feral</td>
<td>felicity</td>
<td>foppish</td>
<td>fractious</td>
<td>gnostic</td>
</tr>
<tr>
<td>histionics</td>
<td>iniquitous</td>
<td>interdiction</td>
<td>intransient</td>
<td>jingoist</td>
</tr>
<tr>
<td>luddite</td>
<td>misandry</td>
<td>misogyny</td>
<td>modus vivendi</td>
<td>nihilism</td>
</tr>
<tr>
<td>ostensibly</td>
<td>pique</td>
<td>perspicacious</td>
<td>praetorianism</td>
<td>proselytize</td>
</tr>
<tr>
<td>raison d'être</td>
<td>red herring</td>
<td>refugium</td>
<td>reify</td>
<td>sartorial</td>
</tr>
<tr>
<td>satorii</td>
<td>scatological</td>
<td>screed</td>
<td>seditious</td>
<td>senescence</td>
</tr>
<tr>
<td>sine qua non</td>
<td>soliciious</td>
<td>solipsism</td>
<td>sui generis</td>
<td>tautology</td>
</tr>
<tr>
<td>troglodyte</td>
<td>ubiquitous</td>
<td>vassalage</td>
<td></td>
<td>teleology</td>
</tr>
</tbody>
</table>

2.9. References and data sources

The definition of “big words” and so on are taken from the following dictionary.


The familiarity ratings that defined the category types for Figure 2.1, and the numbers in the left panel of Figure 2.2 and in Table 2.1 and Figure 2.2 are from the *Hoosier Mental Lexicon.* As noted earlier, the *HML* is a list of about 19,300 English
words taken from several small dictionaries such as the *Webster's Pocket Dictionary*. It was developed by David Pisoni and his colleagues at Indiana University so that psychologists could use it. Along with the spelled form of each word, the HML lists the average familiarity rating (as described in Table 2.1) and the frequency of the word in the Kučera and Francis (1967) sample of English texts, as well as several other useful bits of information that we’ll talk about in another chapter. The following papers are the most detailed description of how the familiarity ratings were collected and the earliest journal articles that describe the *Hoosier Mental Lexicon* and make generalizations from the numbers.


The Kučera and Francis (1967) sample was described in the following book.


The numbers in the center panel of Figure 2.1 and in Figure 2.3 are from:


The numbers in the right panel of Figure 2.1 and in Figure 2.4 are from:


The *Moe, Hopkins, and Rush* (1982) wordlist used in exercise 6 is taken from a study of words that first graders might know. This is a list of 6366 distinct words that were transcribed from recordings of conversations with first graders in Nebraska. To get pronunciations, Melissa Epstein looked up each word in the CMU pronouncing dictionary at http://www.speech.cs.cmu.edu/cgi-bin/cmudict. For words not in the CMU dictionary, she transcribed her own pronunciation using the same ASCII variant of the IPA. The following book described the study and gave the list of words.

Appendix 2A. Some fun facts about word history

The expressions “big word” and “small word” are both quite old, but the expression “four-letter word” seems to be much more recent, as you can see from the following definitions and citations in the Oxford English Dictionary (online).

Here are the OED definition and citations for sense II.9 of big.

II. Having great effects, importance, distinction, etc.

9. Of a word or phrase: intended or tending to impress, overawe, or confuse the hearer through being drawn from scholarly or elevated vocabulary.

1561 T. NORTON tr. J. Calvin Inst. Christian Relig. III. xi. f. 175, Osiander thinketh that with this so childishe a cauillation he hath gotten all thinges, he swelleth, he leapeth for ioye, and stuffeth many leaues full with his bigge wordes. 1630 P. MASSINGER Renegado I. iii. sig. C3, For all your bigge words, get you further off. 1856 R. MCWARD Επαγωνισµοι(1723) 356 Cloathed and adorned with the Busk and Bravery of beautiful and big Words. 1705 S. WHATELEY in W. Perry Hist. Coll. Amer. Col. Ch. I. 167 To be bugbear’d out of our senses by big words. 1872 ‘M. TWAIN’ Roughing It (1972) xxvii. 190 His Partingtonian fashion of...using big words for their own sakes. 1920 Amer. Woman Aug. 8/2 But really, Robert needn't be so superior with his big words. 1946 Nature 24 Aug. 252/2 It is high time we began to try to find out what this is instead of mouthing big phrases such as ‘psychobiologic unit’. 1998 K. LETTE Altar Ego (1999) xi. 108 See, that's one of the reasons I like yer. 'Cause yer know all them big words, an' 'cause yer one of the sweetest women I've ever had the pleasure to suck.

The OED defines four-letter within a longish list of “Special combinations” under word class C, where it is followed immediately by the phrase four-letter man. From the citations given, the latter would seem to be the older term.

C. attrib. and Comb.

four-letter a., consisting of four letters; applied esp. to any of several monosyllabic English words, referring to the sexual or excretory functions or organs of the human body, that are conventionally excluded from polite use; four-letter man, an obnoxious person;

1923 J. MANCHON Le Slang 265 Shit...un type embêtant...L'euph. est *four-letter-man. 1924 J. SUTHERLAND Circle of Stars xxiii. 236 Carter isn’t that kind of a four letter man if he does soak. 1927 C. S. LEWIS Let. 12 Dec. (1966) 122 Louis the Pious was ‘a man of blameless and virtuous habits’tho’ every other sentence in the chapter makes it clear that he was a four letter man. 1934 Amer. Speech IX. 264/1 The obscene
‘four-letter words’ of the English language are not cant or slang or dialect, but belong to the oldest and best established element in the English vocabulary. *Ibid.* 267/1 For most people, the bare word forms of these four-letter words have become sexual fetishes. 1935 E. HEMINGWAY *Green Hills Africa* (1936) II. iii. 97 Ashamed at having been a four-letter man about books. 1947 A. HUXLEY *Let.* 9 Mar. (1969) 568 She would bring him to amorous life again by re-assuming her cockney accent... going very nearly to the point of murmuring four-letter words into his ear. 1960 *Times* 7 Nov. 17/4 Having regard to the state of current writing, it seems that the prosecution against *Lady Chatterley* can only have been launched on the ground that the book contained so-called four-letter words. 1962 I. MURDOCH *Unofficial Rose* xvii. 164 Felix regarded Randall as a four-letter man of the first order. 1969 N. COHN *AWopBopaLooBop* (1970) xx. 191 He was heckled. Immediately, he exploded in a rash of four-letter words and the curtain came down.

Here are the OED definitions and citations for sense II.3.d of *small.*

d. Of words: Short, simple. †Also of language: Simple, plain.

c1250 *Gen. & Ex.* 18 Dan man hem telld soðe tale Wid londes speche and wordes smale. 1679 V. ALSOP *Melius Inq.* I. iii. 135 As if we were not as much obliged to tell the People their duty as God our wants in small English. 1821 BYRON

*Sardan.* I. ii. 511 Your first small words are taught you from her lips.

Note that the last of these citations is from the following exchange between King Sardanapalus and his female slave Myrrha, in Act I, Scene iii, of Lord Byron’s play *Sardanapalus,* which was published in 1821:

*Sar.* Save me, my beauty! Thou art very fair, And what I seek of thee is love—not safety.  
*Myr.* And without love where dwells security?  
*Sar.* I speak of woman’s love.  
*Myr.* The very first  
Of human life must spring from woman’s breast,  
Your first **small words** are taught you from her lips,  
Your first tears quenched by her, and your last sighs  
Too often breathed out in a woman’s hearing,  
When men have shrunk from the ignoble care  
Of watching the last hour of him who led them.