

with subfields that correspond to the tables (e.g., `b.nodes.x`, `b.nodes.y`, and so on). Another function, `drawbridge`, will draw the truss, with appropriate labeling.

```
>> drawbridge(b);
```

A third program, `findtensions`, computes the stresses in the truss members. It returns a bridge structure that has all the same information as the structure given as an argument but has the `.stress` and `.supportforces` fields filled in.

```
>> bt = findtensions(b);
```

`Drawbridge` can use this information to color-code the drawing.

```
>> drawbridge(bt);
```

**Exercise 11.13:**

Write a program `memberCoords` that takes two arguments, a bridge structure (as constructed from files by `readbridge`) and a member number and returns two vectors, one giving the  $x$ -coordinates of the endpoints and the other the  $y$ -coordinates.

Write another function, `drawMembers`, that loops over all of the members and, using `memberCoords`, draws the whole bridge.

**Exercise 11.14:**

Using the output of `findtensions`, expand `drawMembers` to include graphical depictions of the tension information. In Chapter 17, we will show the techniques for writing `findtensions`.

**Exercise 11.15:**

Make a movie of the changing stresses on bridge elements as a load is moved across the bridge. Note that when a load is applied to the member between two nodes, the load is split between the two nodes. If the load is situated a fraction  $p$  of the distance from node A and node B, then node A carries fraction  $1 - p$  of the load and node B carries fraction  $p$ . (*Hint:* Write a function that takes as an argument time  $t$ , computes the location of the load, and fills in the load’s part of the database structure. Then use `findtensions` and `drawbridge` in the ordinary way.)

## 11.9 Project: Random Generation of Text

A simple model of text, proposed originally by the information theorist, Claude Shannon, looks at the probabilities of words. As we saw in Section 11.6, some words occur more frequently than others. By tabulating these frequencies, we capture some of the structure of the text. By generating text according to the frequencies, we can help to see whether this frequency-based structure indeed captures important aspects of the text.

To illustrate, we can read in a body of text, word by word. Here is the novel *Moby Dick*:

```
>> w = textread('mobydick.txt', '%s', 'delimiter', '.,:;=¿"!?'');
>> length(w)
ans: 215743
```

To generate the random text, we can simply pull words at random out of this corpus. In so doing, we will follow statistically the frequency of words in the text. One way to sample randomly is to generate a set of random integers, each between 1 and the length of the text. For example, to sample 100 words at random, we generate 100 random integers:

```
>> inds = ceil( length(w) * rand(100,1) );
```

Here are the sampled words:

**of ashore stood his of even I blackness heads that gin a sea take ladder that Mungo porcupine Fates soggy his again the the into in Jonah great town To all pitched look side ll months to wonder nothing mattress tell the by manliest picture sighs have and the the tapping where as prepares not do array his darkness opinions the belongs and as difficulty the his ship glides wide expostulations curious his previous shadows all part up no monstrous the with to in seemed of hand to I out has rather sees tempest place Thirty But**

A rough read! Randomly ordered words do not seem much like real text.

Shannon knew this. He proposed a model of text that takes into account the fact that the probability of a word depends on its context. For example, it's very unlikely to have pairs of words such as “the his” or “the the,” but both of these pairs appear in the preceding text. In Shannon's model, one looks at the “conditional probability” of each word given the previous word.

Going through the list of words in *w*, we can pull out every word that occurs after the word “the.” It turns out that there are 14,302 of them in *Moby Dick*. Here is part of the ASCII-ordered list:

```
21st act adulterer affrighted afternoon age ages
aggregated air air air ... world world world worm
wound wrinkled yawning year young
```

As anticipated, the phrase “the the” does not ever appear in the text. However, phrases like “the air” and “the world” do appear several times; perhaps this is to be expected in a novel involving a sailing ship.

The conditional probabilities contain additional information about the text. We can generate text that obeys these probabilities with the following algorithm. To generate  $N$  random words from the text,

1. Start with an initial word, called *iword*.
2. Construct a list of all the words in the text that follow *iword*.
3. Choose at random one of the words from this list, calling it *rword*.
4. Set *iword* = *rword* and repeat until  $N$  words are generated.

The sequence of  $N$  *iwords* constitutes the text.

Here is an example from *Moby Dick*, starting with “call” as an initial word. The superscripts indicate how many choices there were when each word was selected at random:

call<sup>0</sup> all<sup>55</sup> the<sup>1477</sup> fishery<sup>14302</sup> in<sup>60</sup> others<sup>4136</sup> that<sup>38</sup>  
 strange<sup>2937</sup> town<sup>96</sup> some<sup>17</sup> of<sup>614</sup> oil<sup>6579</sup> so<sup>78</sup> day<sup>1041</sup>  
 grew<sup>158</sup> out<sup>14</sup> more<sup>521</sup> and<sup>502</sup> every<sup>6358</sup> roll<sup>230</sup> of<sup>17</sup> it<sup>6579</sup>  
 or<sup>2375</sup> the<sup>708</sup> whales<sup>14302</sup> these<sup>264</sup> things<sup>397</sup> such<sup>127</sup> an<sup>366</sup>  
 african<sup>589</sup> elephant<sup>3</sup> vast<sup>15</sup> host<sup>73</sup> would<sup>10</sup> not<sup>427</sup>  
 love<sup>1131</sup> steelkilt<sup>24</sup> hailed<sup>35</sup> him<sup>20</sup> therefore<sup>1046</sup> you<sup>66</sup>  
 be<sup>861</sup> social<sup>1038</sup> meeting<sup>12</sup> a<sup>10</sup> weaver's<sup>4630</sup> loom<sup>1</sup> are<sup>8</sup>  
 scattered<sup>595</sup> people<sup>9</sup>

This is much easier to read; all of the word-to-word transitions actually occur in the original text.

An even stronger model of text comes from defining the context for each new word to be the previous two words. To generate text according to this model, we start with two initial words, generate a third word, and then iterate. For example, the following text was generated starting with “Call” and “me.” The third word, “an,” happens to be one of three in the text that follows “Call me.” The next word generation occurs in the context of “me Ishmael.”

call<sup>0</sup> me<sup>0</sup> an<sup>3</sup> immortal<sup>3</sup> by<sup>1</sup> brevet<sup>1</sup> yes<sup>1</sup> there<sup>1</sup> is<sup>1</sup>  
 again<sup>83</sup> carried<sup>1</sup> forward<sup>1</sup> the<sup>1</sup> boat<sup>3</sup> and<sup>104</sup> grievous<sup>28</sup>  
 loss<sup>1</sup> might<sup>1</sup> ensue<sup>1</sup> to<sup>1</sup> nantucket<sup>1</sup> and<sup>3</sup> come<sup>10</sup> to<sup>6</sup>  
 think<sup>34</sup> of<sup>18</sup> a<sup>34</sup> baby's<sup>333</sup> pulse<sup>2</sup> and<sup>1</sup> lightly<sup>2</sup> say<sup>1</sup>  
 of<sup>1</sup> it<sup>1</sup> from<sup>95</sup> an<sup>13</sup> unquestionable<sup>1</sup> source<sup>1</sup> bear<sup>1</sup>  
 with<sup>1</sup> me<sup>2</sup> i<sup>24</sup> only<sup>23</sup> wish<sup>4</sup> that<sup>1</sup> we<sup>2</sup> seemed<sup>28</sup> to<sup>1</sup>  
 drink<sup>57</sup> a<sup>7</sup> bottle<sup>1</sup> of<sup>2</sup> bordeaux<sup>1</sup> he<sup>1</sup> added<sup>67</sup>  
 motioning<sup>5</sup> to<sup>1</sup> me<sup>1</sup> though<sup>89</sup> indeed<sup>2</sup> i<sup>9</sup> might<sup>3</sup>  
 proceed<sup>9</sup> with<sup>1</sup> a<sup>2</sup> globular<sup>247</sup> brain<sup>1</sup> and<sup>1</sup> heart<sup>5</sup> and<sup>3</sup>  
 hand<sup>8</sup> backed<sup>1</sup> by<sup>1</sup> a<sup>2</sup> heedful<sup>82</sup> closely<sup>1</sup> calculating<sup>1</sup>  
 attention<sup>1</sup> to<sup>1</sup> that<sup>4</sup> not<sup>53</sup> an<sup>10</sup> unfair<sup>4</sup> presumption<sup>1</sup>  
 i<sup>1</sup> say<sup>1</sup> all<sup>70</sup> of<sup>2</sup> ye<sup>25</sup> nor<sup>23</sup> can<sup>1</sup> whalemens<sup>5</sup> be<sup>1</sup>  
 recognised<sup>1</sup> as<sup>1</sup> the<sup>1</sup> ship<sup>212</sup> the<sup>193</sup> ship<sup>17</sup>

Compare this text to the original on page 100.