Appendices

to the book:

Who Wrote the Bible Code?

by Randall Ingermanson

Appendices A through D are intended for technical readers who would like to verify the correctness of the results reported in my book *Who Wrote the Bible Code?* My intent is that the main ideas of the book should be understandable without reading these appendices. The math to be presented here should make sense to someone with an undergraduate degree in math, science, engineering, or related disciplines.

Appendix A: Statistics

Elementary Statistics

First, we'll discuss some elementary statistics. A standard reference for this is *Numerical Recipes in C: The Art of Scientific Computing*, Second Edition, by William H. Press, et. al., Cambridge University Press, (1992). The following summarizes the discussion in Chapter 14 of that book.

Suppose we are measuring some quantity x. Generally, each measurement will have some small error, so we'll make more than one measurement. If we have a set of N measurements $\{x_1, x_2, \ldots, x_N\}$, we define the mean:

$$\bar{x} \equiv \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \text{(A1)}$$

We define the standard deviation

$$\Delta x \equiv \sqrt{\sum_{i=1}^{N} \frac{\left(x_i - \bar{x}\right)^2}{N-1}} \qquad \text{(A2)}$$

In the text, we'll generally call this number the "spread." We'll denote the actual value of x by $\langle x \rangle$.

We'll make the conventional *assumption* in this book that, for the statistical ensemble of all sample sets of size N, the mean $\bar{x}$ is a random variable with a distribution that is approximately Gaussian, with a mean of $\langle x \rangle$ and a standard deviation

$$\sigma = \frac{\Delta x}{\sqrt{N}} \tag{A3}$$

We'll refer to this in the text as the "Law of Large Numbers."

When comparing two members of this statistical ensemble, that is, two means $\overline{x_1}$ and $\overline{x_2}$, each with a standard deviation Δx_1 and Δx_2, the difference between the two means is a random variable with a composite standard deviation given by:

$$\Delta x_{\text{diff}} = \sqrt{\Delta x_1^2 + \Delta x_2^2} \tag{A4}$$

If the sample sizes N_1 and N_2 are less than about 30, then this difference is governed by Student's t-distribution. For larger samples, this distribution converges to Gaussian statistics. We'll always work with samples large enough to justify using a Gaussian distribution.

Binomial and Gaussian Statistics

A good reference for binomial statistics is *Fundamentals of Statistical and Thermal Physics*, by Frederick Reif, McGraw-Hill, Inc. (1965). The following is taken from Chapter 1 of that book.

Consider a simple random experiment that can have two outcomes. An example is flipping a coin. We don't assume that the coin is fair; it may be specially designed to come up

heads more often than tails. We'll denote the probability of heads by some value p between 0 and 1. The probability of tails is therefore $1 - p$.

If we conduct an experiment in which we flip the coin N times, we'll observe some frequency of heads F. Note that F must be an integer between 1 and N.

Now flipping a coin is a random process, so we can't expect to predict the value of F for our experiment. However, if we conducted an infinite number of such experiments, then the mean value of F would converge to a quantity $\langle F \rangle$, the "expectation value" of F. This expectation value can be computed exactly, and its value is

$$\langle F \rangle = Np \tag{A5}$$

We can also calculate the expected value of the standard deviation of F exactly:

$$\langle \Delta F \rangle = \sqrt{Np(1 - p)} \tag{A6}$$

Now any experiment of N coin-flips will yield not only a frequency F, but also an estimate P of the probability p. That estimate is given by:

$$P = \frac{F}{N} \tag{A7}$$

The expectation value of P is p, of course. We can also calculate the expectation value of the standard deviation of P. It's convenient to denote this by Δp:

$$\Delta p \equiv \langle \Delta P \rangle = \sqrt{\frac{p(1 - p)}{N}} \tag{A8}$$

For binomial statistics, one can show that, for N large, P is approximately normally distributed about p—that is, it's governed by Gaussian statistics. It's convenient to normalize a Gaussian distribution. Here, we do so by assigning a value Z to each measurement P, as follows:

$$Z \equiv \frac{P - p}{\Delta p} \tag{A9}$$

Z is informally called the "Z-score" in the main text of this book. There, we don't distinguish between Z-scores for the distributions of *P* and *F*, since these distributions are related by a trivial transformation. Their Z-scores are therefore identical.

One can estimate the statistical significance of a given value of *Z* using the complementary error function erfc(*Z*), which measures the probability that a random process could yield a Z-score as large in magnitude (and with the same sign) as the observed *Z*:

$$\text{erfc(Z)} \equiv \frac{1}{\sqrt{2\pi}} \int_Z^\infty \exp\left(-\frac{t^2}{2}\right) dt \qquad \text{(A10)}$$

An implementation of erfc is given in *Numerical Recipes*. Spreadsheets also often provide this function. In most scientific and statistical work, *Z* is considered significant if its absolute value is greater than 2 or 3. In this book, we'll be looking for *extreme* statistical significance. The reason is that we're searching for phenomena that would have extraordinary implications. Therefore, we have a right to demand an extraordinary level of evidence.

There is no commonly accepted definition of "extreme statistical significance." I'll simply define it here to be a *Z* with absolute value greater than 5.

Poisson Statistics

An interesting special case of binomial statistics occurs when *p* approaches zero. This would be the case with a very unfair coin! Of course, binomial statistics apply to other experiments besides coin-tosses. An example would the probability of finding a misspelled word in some given textbook.

We again consider an experiment with *N* trials, and now we denote the expected frequency by λ, which is the conventional notation for Poisson statistics:

$$\lambda = Np \tag{A11}$$

We're interested in the case where p is small. One can then show that an approximation for the probability $W(k)$ of observing k events in N trials is:

$$W(k) = \frac{\lambda^k}{k!} e^{-\lambda} \tag{A12}$$

This probability distribution will be very useful in studying the probability tables of two-letter clusters — "digrams" — and three-letter clusters — "trigrams" — in natural languages. In English, the most frequent digram is "th," which occurs with a probability of about 3.7%. Most digrams and all trigrams occur much less frequently than this, and so Poisson statistics provide a useful tool for analyzing them. We discuss digrams and trigrams at length in Chapters 7 through 12.

Appendix B: Chi-squared Analyses

Definition of Chi-squared Statistics

In Appendix A, we discussed random experiments in which each trial could yield one of two possible outcomes. An example is flipping a coin. Here, we'll generalize those ideas to experiments in which a trial can yield one of several possible outcomes. An example is dice rolling. Most of these ideas are explained quite well in Section 14.2 of *Numerical Recipes,* cited in Appendix A. We'll develop the equations first for dice, and then generalize.

Each die has six sides. If the dice are fair, then any of the sides will come up with equal probability. If the dice are not fair, then each number will come up with some characteristic probability. We'll denote by p_1 the probability of a given die coming up 1, and so on for 2 through 6. Whether a die is fair or not, the six probabilities p_i satisfy a constraint:

$$\sum_{i=1}^{6} p_i = 1 \qquad \text{(B1)}$$

We say that the 6 probabilities p_i have 5 "degrees of freedom," since we could use any 5 of them to determine the sixth.

As in Appendix A, we could perform an experiment by rolling a die N times and recording the observed frequencies F_i for each of the six possible outcomes. From the frequencies, we then compute estimates of the probabilities:

$$P_i = \frac{F_i}{N} \tag{B2}$$

Just as in the case of the binomial distribution, these will be randomly distributed with a mean of p_i and a standard deviation:

$$\Delta p_i \equiv \langle \Delta P \rangle = \sqrt{\frac{p_i(1 - p_i)}{N}} \tag{B3}$$

For each measured probability P_i we can define a Z-score which measures the normalized deviation of the observed value from the expected value:

$$Z_i \equiv \frac{P_i - p_i}{\Delta p} \tag{B4}$$

Unfortunately, these are six numbers, and we'd prefer to work with only one, a number representing the aggregated deviation of all observed values from their respective expected values. Also, we'd like to have some function like erfc that allows us to determine the statistical significance of our results.

The well-known solution is to define the chi-squared statistic:

$$\chi^2 \equiv \sum_{i=1}^{6} Z_i^2 \tag{B5}$$

If any of the quantities Z_i is large, then chi-squared gets very large. Now notice that if the Z_i's are distributed according to Gaussian statistics, then the expected contribution of each term should be 1. Actually, because of the constraint (B1), the expected contribution is a bit less than

1. The net result is that the expectation value of chi-squared is just about equal to the number of degrees of freedom, which we'll call *D* from here on.

Chi-squared can be larger than *D*, but we don't expect it to be very much larger. We need a way to quantify this statement, so that we can assign statistical significance to a chi-squared analysis.

Measuring the Statistical Significance of Chi-squared

Just as the statistical significance of *Z* could be determined using the function erfc, the statistical significance of chi-squared can be determined using the incomplete gamma function. As explained in Section 6.2 of *Numerical Recipes*, there are actually two functions that go by this name, and they're complementary to each other. The one we mean is defined as:

$$Q(a,x) \equiv \frac{1}{\Gamma(a)} \int_x^\infty e^{-t} t^{a-1} dt \qquad \text{(B6)}$$

Suppose we've done a chi-squared analysis with *D* degrees of freedom and have computed a value χ^2. For a random process, the probability of observing a value as large as the one we measured is given by:

$$Q\left(\frac{D}{2}, \frac{\chi^2}{2}\right) \qquad \text{(B7)}$$

As with erfc, the function Q is implemented in *Numerical Recipes* and in many spreadsheet programs. There is an error in the algorithms printed in both the first and second editions of *Numerical Recipes*. I discovered the error early on in my testing of the software for this book. The problem was a failure to check convergence of a series before terminating it. My

software contains the correct algorithm; let the reader beware who trusts blindly in the printed recipe!

A more important point is that the function Q gives an accurate estimate of the statistical significance of chi-squared only if the standard deviations Δp_i have been accurately estimated. As discussed in *Numerical Recipes*, good error estimates are not always easy to come by. This can cause a poor estimate of the statistical significance.

For example, if these standard deviations are systematically off by 10%, then the value of chi-squared will be systematically off by about 20%. For a large number of degrees of freedom, this will translate into a fairly serious error in the estimate of statistical significance.

Because of this problem, we'll use a different method of estimating statistical significance in this book, one that allows us to correct for systematic errors very accurately. The method is simple enough that we've explained it in Chapter 11. The idea is to regard the chi-squared statistic itself as a random variable. By doing a large number of chi-squared analyses, we can determine the mean and standard deviation of this statistic for two different populations. Then we compare them by calculating a composite Z-score for the difference of the two means, as discussed in Appendix A. The composite Z-score supplies us with an estimate of statistical significance. Any systematic errors in the standard deviations used in the chi-squared analyses *will apply to both populations.* So taking the difference effectively eliminates the systematic errors.

This method works much better than using the incomplete gamma function does.

Generalizing Chi-squared

The formulae above were motivated by the example of rolling dice. It is trivial to generalize them to an arbitrary random process with multiple outcomes. If there are *n* possible outcomes, then we define the probabilities p_i where the index now ranges from 1 to *n.*

The constraint is now:

$$\sum_{i=1}^{n} p_i = 1 \tag{B8}$$

and the number of degrees of freedom *D* becomes:

$$D \equiv n - 1 \tag{B9}$$

The chi-squared statistic generalizes to:

$$\chi^2 \equiv \sum_{i=1}^{n} Z_i^2 \tag{B10}$$

In this book, we'll apply a chi-squared analysis to the observed probability tables of digrams and trigrams. For a text with known letter probabilities, we can compute the exact probabilities of randomly choosing any given digram or trigram. The chi-squared analysis then lets us compare the exact theoretical probabilities to the actual observed probabilities. Some technical issues are discussed in Appendix D.

Appendix C: Entropy

Definition of Entropy

Entropy is a widely used concept in information theory. My favorite reference on entropy and information theory in general is *An Introduction to Information Theory: Symbols, Signals and Noise*, by John R. Pierce, Dover Publications, Inc., (1980). This book covers the topic at a popular level and discusses many interesting ideas on language, including lots of information on digrams and trigrams.

We'll define entropy first, and then explain why this weird looking definition is actually useful. Consider a message encoded using a set of n symbols. Measure the frequencies at which each of the symbols occurs in the text, and divide by the number of symbols in the text to compute a set of observed probabilities P_i. Then the entropy h is defined as:

$$h = -\sum_{i=1,\, P_i \neq 0}^{n} P_i \log_2 P_i \qquad \text{(C1)}$$

Let's discuss this formula in detail.

First, why the minus sign? Note that all of the P_i's are less than or equal to 1. Therefore, their logarithms are all negative. We'd like entropy to be a positive number, so we throw in a leading minus sign.

Second, why the restriction $P_i \neq 0$ in the sum? L'hopital's Rule implies that in the limit of $P_i \to 0$, the contribution to the sum vanishes.

Third, what's the meaning of it all? The short answer is that entropy counts the minimum number of bits per symbol required to encode the message. If the message is highly redundant, then not many bits are needed since we could find a way to encode the message without so many symbols. If the message is completely random, then we have to encode every single symbol, and the entropy is then just a measure of how many bits are required to encode a single symbol.

Let's consider some examples.

Suppose our message is a report of the results of a coin flipping experiment. We could let 0 represent tails and 1 represent heads. The message might look something like this:

01110010000110101110110001100100010010110101010110101 00

If the coin is fair, then we should see equal numbers of heads and tails. Then

$$P_0 = P_1 = \frac{1}{2} \qquad \text{(C2)}$$

It's trivial to calculate the entropy in this case: $h = 1$.

Since we have only two symbols, 0 and 1, we can represent each symbol with only 1 bit. We see that in this case, as we claimed, the entropy is equal to the number of bits required per symbol.

Now suppose the coin were biased. Suppose it comes up heads only once in a million times. A piece of the message might look like this:

00000000000000000000000000000000000000000000000000000000

Do we really need to encode each one of these 0's? No! We could instead just encode the number of 0's between each 1. That would be much more efficient, and would lose no information. You can easily dream up an infinite number of possible ways to encode the data. The entropy is easily calculated using (C1): $h = 2.14 \times 10^{-5}$. That is, the message requires only 21.4 bits per million symbols in order to encode the complete message with no loss of information. The entropy is low and the redundancy is high.

Now suppose we have an alphabet with *n* characters, and our message is created by randomly choosing characters. (For example, a monkey playing on a typewriter.) Since each character has an equal probability of appearing at any point in the message, we have:

$$P_i = \frac{1}{n} \tag{C3}$$

Then we immediately find the entropy:

$$h = \log_2 n \tag{C4}$$

For a 2-letter alphabet, the entropy is 1. For a 4-letter alphabet, the entropy is 2. For an alphabet with 2^n characters, the entropy is *n*. In each case, the entropy is the number of bits needed to make the simplest possible binary encoding of the alphabet.

For extra credit, you can use the calculus of variations to show that (C4) is the maximum entropy possible, only achieved when (C3) holds. (Hint: account for the constraint (B8) using a Lagrange multiplier.)

Application to Real Texts

The digits of π should be randomly distributed among the ten symbols 0 through 9. So we expect the entropy of a text containing the digits of π to be:

$$h_\pi = \log_2 10 = 3.321928... \quad \text{(C5)}$$

In fact, this is very close to the entropy we actually measured in Chapter 8 on the first fifty thousand digits of π. The measured value is 3.321842. The discrepancy is due to the fact that the digits aren't exactly equally distributed. In a random text, there is bound to be a little variation, as we discussed in Appendices A and B. The variation is at about the level we expect, so there's no cause for alarm.

For the English alphabet, which has 26 characters, we expect the entropy of a completely random text to be

$$h_{\text{random}} = \log_2 26 = 4.7004397... \quad \text{(C6)}$$

This is *not* what we actually measure for any real, meaningful English text. English, like every language, has some redundancy, which lowers the entropy from the maximal value (C6). For a typical adult-level English text, the entropy computed using the observed letter-probability table is about 4.14. A text like *The Cat in the Hat,* which has a reduced vocabulary, will have a smaller entropy.

When the symbols being studied are digrams and trigrams, there are some extra wrinkles.

First off, if we want to compare apples to apples, the formula for entropy (C1) needs to be divided by a factor of 2 (for digrams) or 3 (for trigrams). The reason is that entropy measures bits per symbol, but we want to compare bits per character. So we need to divide by characters per symbol. This allows us to do a meaningful comparison of the digram and trigram entropies to the letter entropy.

Second, as we discussed in Chapters 8 and 9, natural languages have digram and trigram probability tables that can't be computed naively from the letter probabilities.

For example, the digram “qe” is much less common than the digram “qu,” even though the letter *e* is much more common than the letter *u*.

As another example, the trigram “eee” is quite rare in English, even though it “ought” to be the most common trigram, based on the letter probability tables (since *e* is the most common letter).

Because of this, when you compute the entropy of a text based on the digram probability table, you get a smaller entropy (corresponding to higher redundancy) than you do with the letter probability table. Likewise, the trigram table gives an even smaller result for the entropy.

Third, there’s an even subtler effect that shows up very clearly when you look at short texts. We mentioned this in passing in Chapter 10, and promised an explanation in the appendices. Here is that explanation.

Finite Size Text Effects

Imagine taking an infinitely long English text and measuring its entropy using the letter probabilities. We know that the digram table would give a different entropy, because English has correlations between letters in the digrams. Suppose we scramble this infinite text, eliminating all these correlations. Now the digram table is exactly what you’d predict based on the letter probabilities, and the digram entropy is identical to the letter entropy.

This does not hold true for a finite text.

Since we have 26 letters, there are 676 possible digrams and 17,576 possible trigrams. We haven’t changed the letter probabilities, so letters like *q* and *z* are still pretty rare. That

means that digrams like “qq”, “qz”, and “zz” are also rare, even in a fully randomized text. The same can be said of trigrams like “qqq.”

Say the original text is only a thousand characters long. Then we’ll observe only 999 digrams and 998 trigrams. This is true of both the original text and the randomized text. Some digrams probably won’t appear at all, and *most* trigrams won’t.

This means that we’ll observe zero probabilities for certain digrams and certain trigrams. It’s not that they *actually* have zero probability. The problem is that we aren’t looking at enough text to measure a nonzero probability.

But remember that formula (C1) excludes any symbols with a measured probability of zero. So those digrams and trigrams won’t contribute to the entropy.

It’s possible to estimate this effect very accurately. In what follows, I’ll refer only to digrams, but the same analysis applies also to trigrams. We consider the statistical ensemble of all possible randomized texts of a given length with a given letter probability table. Now we ask what is the expectation value of the digram entropy we should observe.

The first step is to use the results of Appendix D to compute the theoretical probabilities of all possible digrams. We’ll denote these probabilities by p_i. If a given randomized text actually has N digrams in it, then the expectation value of the frequency of each digram is given by:

$$\lambda_i = Np_i \qquad \text{(C7)}$$

It’s important to remember that these are theoretical probabilities and frequencies we’re dealing with here. Real observed frequencies have to be integers. The λ_i’s are not integers, and many of them may be less than 1. The probability that, in a given text, a given digram will occur k times is given by (A12). If a digram occurs k times, then its observed probability P_i will be:

$$P_i = \frac{k}{N} \tag{C8}$$

So we can estimate the expectation value (over the ensemble of randomized texts) of a given digram's contribution to the entropy:

$$\langle h_i \rangle = -\sum_{k=0}^{\infty} W_i(k) \left(\frac{k}{N}\right) \log_2 \left(\frac{k}{N}\right) \tag{C9}$$

Finally, we estimate the expectation value of the entropy of a randomized text:

$$\langle h \rangle = \sum_{i=1}^{N} \langle h_i \rangle \tag{C10}$$

For small values of λ_i, (C9) can be evaluated directly on a computer. For large values, it's smarter to do a power series expansion about the point $k = \lambda_i$, and calculate only the first few terms. It's a bit tedious to compute this expectation value for digrams and trigrams. My software does this calculation. The results agree very well with the actual observed entropies for random texts.

One problem I have not solved is how to compute the standard deviation of the expected entropy. It sounds easy until you try it. Then the mathematics gets a bit ugly.

It's not really necessary to calculate the standard deviation, however. (In fact, it's not necessary to evaluate (C10), either, except as an intellectual exercise to understand why the observed entropies are less than naively expected.) We can simply randomize a large number of texts and measure the mean entropy and the standard deviation. My software does this calculation also, since it's required for the sort of ultra-precise measurements described in Chapters 11 through 14.

Appendix D: Technical Notes on Digrams and Trigrams

Theoretical Probabilities of Digrams and Trigrams

Suppose we're given a text of length N composed using an alphabet having n characters $\{c_1, c_2, ..., c_n\}$. We can measure the frequencies f_i of the n characters and construct the letter frequency table. We then define the letter probabilities p_i:

$$p_i = \frac{f_i}{N} \tag{D1}$$

We often speak of the probabilities and frequencies interchangeably in this book, since we can deduce one set from the other, given N.

We're interested in scrambling the letters of the original text and measuring its properties. Some of these are obvious. Scrambling the text won't change the number of each kind of letter, so the observed letter probabilities P_i of the scrambled text will be identical to those of the original text, p_i.

An important question is what is the spread ΔP_i in the observed probabilities P_i? The answer depends on exactly how we do the scrambling.

One possible process is the following. Choose a letter at random from the text. Then choose a second letter at random from the remaining letters. Continue in this way, choosing letters without replacement until the original pool of letters is empty. At any position in the scrambled text, the probability P_i of finding a given letter c_i is exactly p_i. (This is true, even if the letters in the original text were highly ordered.) Therefore, the spread in the observed letter probabilities of a random text is exactly zero.

That, however, is not exactly the process we want to model. We're interested in studying skip-texts constructed by taking sequences of letters at equal intervals. This process is fairly close to being random, but even at skips of a few thousand characters, we expect one effect to come into play — the fact that the text has a storyline.

For example, compare the first and second halves of Genesis in any English translation. Because Jacob and Joseph play starring roles in the second half of the book, the letter *j* occurs somewhat more frequently in that half. In the section on Abraham, we're going to see the letter *a* just a bit more frequently than usual. So if we're skipping through the Abraham section, our skip-text will have more *a*'s than usual. Later, in the Jacob and Joseph sections, our skip-text will get more *j*'s than normal. So the observed letter probabilities for a given skip-text will depend just slightly on the position within the text. There is no exact way to model this, but we can do an excellent job by ascribing a spread to the letter-probabilities P_i.

As an *approximation*, by analogy to (A8), we'll estimate the spread in the P_i's for skip-texts to be:

$$\Delta P_i = \sqrt{\frac{p_i(1-p_i)}{N}} \qquad \text{(D2)}$$

Next, we want to compute the theoretical probability of randomly choosing a digram. Denote an arbitrary digram as "$c_i c_j$" and denote its theoretical probability by p_{ij}. For a purely random process, we expect:

$$p_{ij} = p_i p_j \tag{D3}$$

Likewise, the theoretical probability $p_{ij...k}$ of randomly choosing an arbitrary sequence "$c_i c_j ... c_k$" is given by:

$$p_{ij...k} = p_i p_j ... p_k \tag{D4}$$

(D3) and (D4) are commonly agreed to be the correct probabilities for uncorrelated sequences of characters. Even for weakly correlated sequences, such as skip-texts at large skips, these probabilities must be correct *on average*.

Now we want to compute the spreads we'll observe for these quantities. Again, the answer depends on which random process we are modeling. For the completely random process we described above, the process obeys binomial statistics. By (A8) the spread in the observed probabilities will be:

$$\Delta p_{ij}^{(\text{random})} = \sqrt{\frac{p_i p_j (1 - p_i p_j)}{N - 1}} \tag{D5}$$

But this doesn't account for the storyline. Because of the slight variations of letter probabilities at different points in the story, there will be just a bit more variation in the digrams. We compute this extra contribution from (D3) using the chain rule:

$$\Delta p_{ij}^{(\text{chain})} = p_i \Delta P_j + p_j \Delta P_i \tag{D6}$$

A similar expression holds for sequences of characters of arbitrary length.

This is all only an approximation to the true situation. We don't fully understand the literary processes which formed the text, and we never will, but we can be sure they were not

random! We'll model the actual spread in the digram probabilities as a combination of (D5) and (D6), and the simplest combination is to just sum the variances:

$$\Delta p_{ij}^{(\text{total})} = \sqrt{\left(\Delta p_{ij}^{(\text{random})}\right)^2 + \left(\Delta p_{ij}^{(\text{chain})}\right)^2} \qquad \text{(D7)}$$

This expression for the spread is used in the chi-squared analyses of digrams and trigrams throughout this book. As we pointed out in Appendix B, small errors in estimating the spread can shift the chi-squared statistic up or down somewhat. The result is that the statistical significance test using the incomplete gamma function is not all that trustworthy. That's why we don't use that test for significance in this book. Instead, we average the chi-squared results for a large number of random texts and a large number of skip-texts prepared in the same way.

That leads to an important question. Exactly how should we prepare our random texts? We'll discuss that next.

Preparing Random Texts

As we saw in the previous section, the fact that our texts may have a storyline can lead to subtle effects. In this book, we're interested in comparing skip-texts to random texts. But what does "random" mean? What randomization process can we use to eliminate the effects of the storyline?

Our procedure is simple. When we randomize a text, we divide it up into blocks of a certain length *B*. (The last block generally will have fewer than *B* characters in it.) Then we randomize each block separately using the fully random procedure mentioned in the previous section. This produces a fairly random text with the same large-scale properties as the original. (Scrambling the English text of Genesis in this way will produce a text with more *j*'s in the

second half than in the first half.) Finally, from this randomized text, we produce a skip-text. This last step tends to produce random texts with the same large-scale properties as the skip-texts of the original.

Most of the experiments in this book study skip-texts with skips from 1 to about 150. For each skip-text, we produce a matching randomized skip-text. This policy effectively cancels out any systematic errors that might creep in as the value of the skip changes.

I have tested the results for different block sizes *B* and found little variation over the range 50 to 1000. In this book, a value of $B = 250$ was used for all calculations shown. In principle, *B* might be considered a free parameter in the analysis, but in fact the results are independent of *B* over a wide range.

A Technical Note on Chi-squared Analyses

In this book, we perform chi-squared analyses to compare the observed probability tables of digrams and trigrams to the theoretical probabilities. Some letters are rare. For example, in English, *q* and *z* are rare. Digrams containing rare letters tend to be very rare. Trigrams containing them tend to be exceptionally rare.

It turns out that many of the digrams and the great majority of the trigrams are so rare that they aren't expected to occur at all in texts of a few tens of thousands of characters. We'd rather not have the chi-squared analysis be dominated by these "unimportant" digrams and trigrams. Therefore, the software has been written to ignore those digrams or trigrams expected to occur less frequently than some threshold value $f_{\min}$, the minimum frequency. For all analyses in this

book, I've chosen f_{min} to be 1. Again, this is an apparent free parameter that turns out not to make much difference in the final results, except to reduce noise.

For entropy calculations, a similar restriction naturally happens by the very definition of entropy. Unless a digram or trigram actually occurs, it doesn't contribute to the entropy. We saw in Appendix C that this fact tends to reduce the measured entropy for small texts. We showed how to estimate this reduction theoretically. Note that in this case, there is no free parameter that could be tuned.