Biological Sequence Analysis:
Probabilistic Models of Proteins and Nucleic Acids

by

Richard Durbin
Sean Eddy
Anders Krogh
Graeme Mitchison
Chapter 1

Background on Probability and Bayesian Statistics

Introduction

This book describes methods for applying probabilistic modeling to biological sequence analysis. We are mainly concerned with determining whether or not a sequence is homologous to a molecule we already know something about. Although ideally the answer to such a question would be yes or no, the stochastic nature of evolution and the complexity of the question makes statistical inference an appropriate approach.

“Homologous” may mean evolutionary, structural, or functional homology, or any combination thereof, depending on whether the molecules in question share common ancestry, common structures, or common biological functions. A similarity between two molecules may be so strong that we can conclude that they share common ancestry, structure, and function. Weaker similarities may make us doubt that the two molecules are functionally homologous though we may still be confident they share common ancestry and structure. Very weak sequence similarities raise the possibility of molecules convergently evolving to similar structures and/or functions without necessarily sharing common ancestry. Functional homology is difficult to infer from computational analysis alone, so we will deal primarily with structural and evolutionary questions.

Computational sequence analysis has been around since the first protein sequences and the first digital computers. Inference of functions and structures by computational analysis has become increasingly important over the
years as our ability to obtain molecular sequence data (particularly DNA sequence data) has outstripped the speed of experimental studies. Excellent textbooks on the subject of computational sequence analysis include Waterman's *Introduction to Computational Molecular Biology* (Waterman 1995) and Sankoff and Kruskal's *Time Warps, String Edits, and Macromolecules* (Sankoff & Kruskal 1983).

To a first approximation, deciding that two biological sequences are similar is no different from deciding that two text strings are similar. The roots of sequence analysis are therefore in computer science, which has provided us with a wealth of algorithms for string comparison and alignment, provided that we can supply a scoring scheme that gives better alignments better scores. Such a scoring scheme might be as simple as “+1 for a match, −1 for a mismatch” and many sequence alignment algorithms have indeed been described in these terms. But biological molecules have evolutionary histories, three dimensional folded structures, and other features which strongly constrain their primary sequence, and these constraints should be taken into account. Therefore, in addition to the mechanics of alignment and comparison algorithms, the scoring system itself requires careful thought. This is more the realm of probability theory rather than computer science. A step forward was Margaret Dayhoff's introduction of probabilistically grounded PAM matrices for scoring pairwise amino acid alignments according to evolutionary preferences for certain substitutions over others. Other aspects of alignment scoring schemes remained the province of trial and error (including the scoring of insertions and deletions).

More sophisticated probabilistic modeling approaches have been brought gradually into computational biology by many routes. Our goal in this book is to bring together these diverse threads and to provide an accessible introduction to the philosophy and methodology of full probabilistic modeling. Full probabilistic modeling methods greatly extend the range of applications that can be underpinned by useful and self-consistent theory. In subsequent chapters, we will describe full probabilistic approaches to pairwise gapped primary sequence alignment, gene finding, multiple sequence alignment, RNA structure alignment, phylogenetic analysis, and protein fold recognition.

The rules that enable one to manipulate probabilities and infer them from data are the topic of this chapter.
1.1 Probabilities and probabilistic models

When we say a model, we mean a parameterized system for assigning a numerical score to an observed data point, where our “data point” is usually a sequence. A probabilistic model is a model in which many or all of the parameters are probabilities. When we say a full probabilistic model, we refer to a model for which the score of a sequence is itself a probability (i.e., the scores of all sequences sum to one). Standard sequence alignments use probabilistic parameters (PAM matrix scores) but also use arbitrary insertion and deletion scoring parameters, and so are not full probabilistic models.

Let us first consider events which have a finite (discrete) set of possible outcomes. A model of a roll of a (possibly loaded) die would have six parameters $p_1 \ldots p_6$, one for each possible roll. The probability of rolling $i$ is $p_i$. To be probabilities, the parameters $p_i$ must satisfy the conditions that $p_i \geq 0$ and $\sum_{i=1}^{6} p_i = 1$. Since our “data” is just a single roll, it is clear that the scores of all possible outcomes also sum to one.

Consider as a second example a model of the set of all sequences of fixed length $M$ with positions numbered $k = 1 \ldots M$. At each position there are 20 possible amino acids that might occur (or 4 possible nucleotides) and these are numbered $i = 1 \ldots 20$. The model has parameters $p_{ik}$ for the probability of a sequence having amino acid $i$ at position $k$. Assuming that residues are independent of each other, the probability of any given sequence of length $l$ is the product of the individual probabilities of its residues. It is easily seen that these probabilities sum to one over all possible sequences. (The probabilities of all sequences that are not length $l$ are zero according to this simple model.)

Another variable we can consider is the type of structural environment that the amino acid finds itself in. Let’s suppose there are $K$ of these, and let’s call them $E_1 \ldots E_K$. Then $P(a_i, E_j)$ denotes the probability of finding, in a protein from the database that we pull out randomly, amino acid $i$ and environment $j$. Again, we expect all these probabilities to sum to 1, so $\sum_{i,j} P(a_i, E_j) = 1$. If we fix $j = j_0$, then $\sum_{i} P(a_i, E_{j_0})$ will not generally be 1. But we can condition on $E_{j_0}$, and obtain the probabilities of amino acids given that the environment is $E_{j_0}$, by defining $P(a_i|E_{j_0}) = P(a_i, E_{j_0})/P(E_{j_0})$ etc.

Probability parameters for a full probabilistic model are typically derived (or learned) from large sets of biological examples. In a certain sense, then, probabilities are used to compactly encode information about the biological
examples. We will discuss later some of the parallels to information theory and encoding.

A probabilistic model is an automatic “Occam’s Razor” (Jefferys & Berger 1992). Occam’s razor is the principle that simple hypotheses should be favored over complex ones. A more complex probabilistic model has more free parameters that can be adjusted to accommodate a wider range of observed data. Because the complex model “spreads itself thin” over the range of possible data – probabilities must sum to one over all possible data – the likelihood that a complex model assigns to any particular data set is lower than the likelihood assigned by a simple model that makes a narrower range of predictions but is also consistent with the data. Hence the simple model is more probable. A more complex probabilistic model must be proportionately even more consistent with the observed data to be favored over a simpler model.

1.2 Bayes’ theorem

As seen above, the probability $P(D \mid M)$ that the observed data $D$ would be obtained from a particular model $M$ is generally straightforward to calculate. We are typically interested in the opposite – the probability $P(M \mid D)$ that a model is the correct one given some observed data. Bayes’ theorem is used to calculate $P(M \mid D)$:

$$P(M \mid D) = \frac{P(D \mid M)P(M)}{P(D)}$$

Bayes’ theorem is at the heart of much of the statistics we will use for full probabilistic modeling. Let us define some terms which we will use often:

- $P(M \mid D)$ is the posterior probability of the model given the data, or just the posterior.
- $P(D \mid M)$ is the probability of the data given the model, also called the likelihood of the model.
- $P(M)$ is the prior probability of the model, or just the prior.
- $P(D)$ is the probability of the data under all models, a normalization term obtained by summing over models: $P(D) = \sum_{M_x} P(D \mid M_x)P(M_x)$. 
Example: an occasionally dishonest casino

Consider a casino that uses two kinds of dice. 99% of the dice are fair. 1% of the dice are loaded so that a six comes up 50% of the time. You pick up a die from a table at random and roll it three times, getting three consecutive sixes. You are suspicious that this is a loaded die. The probability that the die you picked is a loaded one is calculated by Bayes’ theorem:

\[
P(\text{loaded} \mid 3 \text{ sixes}) = \frac{P(3 \text{ sixes} \mid \text{loaded})P(\text{loaded})}{P(3 \text{ sixes} \mid \text{loaded})P(\text{loaded}) + P(3 \text{ sixes} \mid \text{fair})P(\text{fair})}
\]

\[
= \frac{(0.5)(0.01)}{(0.5)(0.01) + (\frac{1}{6})(0.99)}
\]

\[
= 21\%
\]

More sophisticated applications of Bayes’ theorem specify priors over a model’s probability parameters to say that some choices of parameters for the model are a priori more likely than others. Bayes’ theorem allows us to combine prior knowledge with observed data in a principled way. We will return to this topic in detail later in the chapter.

Exercise 1.1 Use the fact that \( P(A, B) = P(A \mid B)P(B) \) to prove Bayes’ theorem.

Exercise 1.2 A rare genetic disease is discovered. Although only one in a million people carry it, you consider getting screened. You are told that the genetic test is extremely good; it is 100% sensitive (it is always correct if you have the disease) and 99.99% specific (it gives a false positive result only 0.01% of the time). Having recently learned Bayes’ theorem, you decide not to take the test. Why? (Hint: Use Bayes’ theorem to calculate the probability that you have the disease if the test comes back positive.)

1.3 Probability Distributions

In the next sections, we will introduce some of the probability distributions we will be needing throughout the book.

So far we have defined probabilities on finite sets. We will also need to consider probabilities on continuous as well as discrete variables. If one has a continuous variable, like the height of person, then the probability that that variable takes a specific value, e.g. that a person’s height is 2 metres (not just 2 metres to the nearest Angstrom, but exactly 2 metres) is zero. But the probability that \( x \) takes a value in some range, \( P(x_0 < x \leq x_1) \) say, can be well-defined and positive. Taking \( x_0 \) to be some lower bound, say \( x_0 = 0 \) for heights, we can regard \( P(x_0 < x \leq y) \) as a function of \( y \); its derivative
with respect to \( y \), \( f(y) = dP(x_0 < x \leq y)/dy \) is called a \textit{probability density} or just a \textit{density}. If we write \( f(y)dy = \delta P(x_0 < x \leq y) \), we can view a density \( f \) as assigning the probability \( f(y)dy \) to the interval \( dy \). A density must satisfy \( f(y) \geq 0 \), all \( y \), and \( \int_\mathbb{R} f(y)dy = 1 \). But note that we can have \( f(y) > 1 \). For instance, the density taking the value 50.5 in the interval \([0, 0.01)\) and the value 0.5 in \([0.01, 1]\) is well-defined on the interval \([0, 1]\).

The distributions of classical statistics, the Gaussian, Poisson, Chi-Squared, and so on (e.g., Keeping 1995), are actually densities, but we shall use the term `distribution’ instead of density for them to adhere to convention.

**The binomial distribution**

The first distribution we consider is possibly the most familiar and most elementary: the binomial distribution. It is defined on a finite set consisting of all the possible results of \( N \) tries of an experiment with a binary outcome, ‘0’ or ‘1’. If \( p_0 \) is the probability of getting a ‘0’ and \( p_1 \) the probability of getting a ‘1’, the probability that \( k \) out of the \( N \) tries yield a ‘1’ is

\[
P(k \text{ ‘}1\text{’ s out of } N) = \binom{N}{k} p_0^{N-k} p_1^k.
\]  

(1.1)

The mean of this distribution is \( N p_1 \) and the standard deviation is \( \sigma = \sqrt{N p_0 p_1} \), see Exercise 1.3.

**Exercise 1.3** Prove that the mean and variance of the binomial distribution are \( \mu = N p_1 \) and \( \sigma = \sqrt{N p_0 p_1} \). Hint: \( \mu = \sum_k k \binom{N}{k} p_0^{N-k} p_1^k \). This can be computed by treating \( p_0 \) and \( p_1 \) as independent variables, differentiating with respect to \( p_1 \), then setting \( p_0 + p_1 = 1 \).

Observe that

\[
\sum k(k-1) \binom{N}{k} p_0^{N-k} p_1^k = p_1^2 \frac{\partial^2}{\partial p_1^2} (p_0 + p_1)^N \bigg|_{p_0+p_1=1} = N(N-1)p_1^2.
\]

**Stirling’s formula and the limit of a binomial**

Consider next what happens as we let \( N \to \infty \). Both the mean and the variance increase linearly with \( N \), but we can rescale to give fixed mean and standard deviation, defining the new variable \( \bar{u} \) by \( \bar{u} = \frac{N}{\sqrt{N p_0 p_1}} \). To calculate the limit of \( \binom{N}{k} \) as \( N \to \infty \) we need a famous approximation for the factorial function called \textit{Stirling’s formula}:

\[
\log n! \approx (n + \frac{1}{2}) \log n - n + \frac{1}{2} \log(2\pi).
\]  

(1.2)

We won’t prove this here, but can show that it is at least plausible by noting that \( \log n! = \sum_{i=1}^n \log i \approx \int_0^n \log x dx = n \log n - n \). This is clearly close to Stirling’s formula.
We can now approximate the binomial coefficient using this formula. After a bit of algebra and neglecting terms in negative powers of $N$, 
\[
\log \left( \binom{N}{k} p_0^{N-k} p_1^k \right) \simeq -\frac{1}{2} \log(2\pi N p_0 p_1) - \frac{w^2}{2}
\]
Exponentiating this, we get:
\[
\binom{N}{k} p_0^{N-k} p_1^k \simeq e^{-\frac{(Nw)^2}{2\pi}} \frac{e^{\frac{w^2}{\sigma^2}}}{\sigma \sqrt{2\pi}}
\]
This is the classic result that, in the limit of a large number of events, a binomial distribution becomes a Gaussian. It can be regarded as a special case of the central limit theorem, which states that the distribution of a sum of $N$ independent random variables, normalised to the same mean and variance, tends to a Gaussian as $N \to \infty$. Consider a single variable taking values '0' or '1' with probabilities $p_0$ and $p_1$, respectively: the distribution of the sum of $N$ copies of this is $P(n) = P(X_1 + \ldots + X_N < n)$, and is precisely the binomial considered above. The central limit theorem states that
\[
\frac{\sum \left(X_i - \frac{N}{2}\right)}{\sqrt{N}}
\]
tends to $\frac{-x^2/2}{\sigma \sqrt{2\pi}}$, which is equivalent to (1.3).

The multinomial distribution

The generalization of the binomial distribution to the case where the experiments have $K$ independent outcomes with probabilities $\theta_i$, $i = 1, \ldots, K$, is the multinomial distribution. The probability of getting $n_i$ occurrences of outcome $i$ is given by:
\[
P(n|\theta) = M^{-1}(n) \prod_{i=1}^K \theta_i^{n_i}.
\]
Here we condition the probability on the parameters $\theta$ of the distribution, which is a natural thing to do in a Bayesian framework, because then the parameters are themselves random variables. In a classical statistics framework the probability of $n$ could, for instance, have been denoted by $P_\theta(n)$. The normalizing constant only depends on the total number of outcomes observed, $|n|$. For fixed $|n|$ it is
\[
M(n) = \frac{n_1! \cdot n_2! \cdot \ldots n_K!}{|n|!} = \frac{\prod n_i!}{|n|!}.
\]
The factorial function is defined only for non-negative integers as $n! = n(n-1)\cdots1$, and $0! = 1$.

For $K = 2$ the multinomial distribution is equal to the binomial distribution.
Example: Rolling a die

The outcome of rolling a die \(N\) times is described by a multinomial. The probability of each of the 6 outcomes is called \(\theta_1, \ldots, \theta_6\). For fair dice where \(\theta_1 = \ldots = \theta_6 = 1/6\) the probability of rolling ten dice and getting five sixes and five ones is:

\[
\frac{10!}{5!5!} \left(\frac{1}{6}\right)^{10} = 4.2 \cdot 10^{-6}.
\]

The Dirichlet distribution

In Bayesian statistics one needs distributions (or more accurately probability densities) over probability parameters to use as prior distributions. A natural choice for a probability distribution over probability parameters is the Dirichlet distribution:

\[
D(\theta | \alpha) = Z^{-1}(\alpha) \prod_{i=1}^{K} \theta_i^{\alpha_i-1}, \quad (1.6)
\]

Here \(\alpha = \alpha_1, \ldots, \alpha_K\), with \(\alpha_i > 0\), are constants specifying the Dirichlet distribution, and \(0 \leq \theta_i \leq 1\). The part of the distribution involving the \(\theta_i\) is the same as for a multinomial distribution. Instead of normalising over the numbers \(n_i\) of occurrences of outcomes, however, one normalises over all possible values of the \(\theta_i\). To put this another way, the multinomial is a distribution over its exponents \(n_i\), whereas the Dirichlet is a distribution over the numbers \(\theta_i\) that are exponentiated. The two distributions are said to be conjugate distributions (Casella & Berger 1990), and their close formal relationship leads to a harmonious interplay in problems such as the MAP estimation of probabilities from counts.

The normalising factor \(Z\) for the Dirichlet defined in (1.6) can be expressed in terms of the gamma function (Berger 1985)

\[
Z(\alpha) = \int \prod_{i=1}^{K} \theta_i^{\alpha_i-1} d\theta = \frac{\prod_i \Gamma(\alpha_i)}{\Gamma(|\alpha|)}. \quad (1.7)
\]

(This integral is only over \(\theta_i\) that sum to one.) The gamma function is a generalization of the factorial function to real values. For integers \(\Gamma(n) = (n-1)!\). For any positive real number \(x\),

\[
\Gamma(x + 1) = x \Gamma(x). \quad (1.8)
\]

It can be shown that the average of the Dirichlet distribution is equal to the normalized parameters, i.e., the average of \(\theta_i\) is equal to \(\alpha_i/|\alpha|\).
For two variables ($K = 2$) the Dirichlet distribution is equal to the more widely known beta distribution, and the normalising constant is the beta function.

**Example: the dice factory**

Consider again our example of a probabilistic model of a possibly loaded die with probability parameters $\theta = \theta_1 \ldots \theta_6$. Sampling probability vectors $\theta$ from a Dirichlet parameterised by $\alpha = \alpha_1 \ldots \alpha_6$ is like a "dice factory" that produces different dice with different $\theta$.

Consider dice factory A with all six $\alpha_i$ set to 10, and dice factory B with all $\alpha_i$ set to 2. On average, both factories produce fair dice – the average of $\theta_i$ is $\frac{1}{6}$ in both cases. But if we find a loaded die with $\theta_6 = 0.5, \theta_1 = 0.1$, it is much more likely to have been produced by dice factory B:

$$
\mathcal{D}(\theta \mid \alpha_A) = \frac{\Gamma(60)}{\Gamma(10)^6} (.1)^{10-1} (.5)^{10-1} \\
= .119
$$

$$
\mathcal{D}(\theta \mid \alpha_B) = \frac{\Gamma(12)}{\Gamma(2)^6} (.1)^{2-1} (.5)^{2-1} \\
= 199.6
$$

The factory with the higher $\alpha$ parameters produces a tighter distribution in favor of fair dice. The sum $|\alpha|$ is inversely proportional to the variance of the Dirichlet. (Don’t be alarmed by the Dirichlet density having a value of 199.6; recall that the values of continuous probability densities at any point may be greater than one.)

A factory that produced almost perfectly fair dice would have very high but equal $\alpha_i$. A factory that produced variably unreliable dice that are still fair on average would have low but equal $\alpha_i$. A factory that reliably produced loaded dice might have $\alpha_6 = 5000, \alpha_1 \ldots 5 = 1000$.

### 1.4 Entropy

Some of the terminology we will use in the book is borrowed from information theory. Information theory has strong connections to probabilistic modeling.

An *entropy* is a measure of the average uncertainty of an outcome. Given probabilities $P(x_i)$ of a discrete set $X$ of $K$ events $\{x_i\}$, the entropy of $\{x_i\}$ is defined by

$$
H(X) = - \sum_i P(x_i) \log(P(x_i))
$$

In this definition, the term $P(x_i) \log(P(x_i))$ is zero for any probability $P(x_i)$ that is zero. Normally we assume that log is the natural logarithm (sometimes written ln). However it is common to use the logarithm base 2 (called
\(\log_2\), in which case the unit of entropy is a ‘bit’. All logarithms are proportional, e.g. \(\log_2(x) = \log_e(x)/\log_e(2)\), so theoretically it does not matter which logarithm is used.

The entropy is maximised when all the \(P(x_i)\) are equal (\(P(x_i) = 1/K\)) and we are maximally uncertain about the outcome of a random sample. If we are certain of the outcome of a sample from the distribution, i.e. \(P(x_k) = 1\) for one \(k\) and the other \(P(x_i) = 0\), the entropy is zero.

**Example: entropy of random DNA**

DNA sequence has four symbols (A,C,G,T). If each symbol occurs equiprobably \((p_i = .25)\) then the entropy per DNA symbol is \(- \sum_{i=A,C,G,T} p_i \log_2 p_i = 2\) bits.

One can think of the entropy as the number of binary yes/no questions needed to discover the outcome. For random DNA, one needs two questions: “purine or pyrimidine?” followed by “A or G?”, for instance.

Entropy also arises as the expected score of the sequences generated by certain probabilistic models when the score is defined to be the log-probability. Suppose, for instance, that the probability of the \(i\)-th residue in some position in a sequence is \(p_i\). Then there is a probability \(p_i\) of score \(\log p_i\), and the expected score is \(\sum_i p_i \log p_i\); namely the negative entropy. The same is true (see Exercise 1.4) when the model defines the probabilities at a set of independent sites.

*Information content* or just *information* is a measure of a reduction in uncertainty after some “message” is received; hence, the difference between the entropy before and the entropy after the message:

\[
I(x) = H_{before} - H_{after}
\]

If the reduction in uncertainty is absolute, i.e. we are told with certainty what the outcome is, \(H_{after}\) is zero and the information is equal to our original uncertainty \(H_{before}\). Thus, entropy is often equated with information. This can be confusing; it leads to the quite counterintuitive view that the more random something is (the higher the entropy), the more information it has. It is not confusing if one thinks of information as a difference in entropy. It is also possible that the reduction in uncertainty is not absolute; there may be noise on the communications channel, for instance, and we may remain somewhat uncertain of the outcome, in which case \(H_{after}\) is positive and the information is less than the original entropy.
Example: information content of a conserved position

Information content is a useful measure of the degree of conservation at a site in a DNA or protein sequence alignment. Say we expect DNA sequence to be random ($p_x = .25$; $H_{base} = 2$ bits), but we observe that a particular position in a number of related sequences is always an A or a G with $p_A = .7$ and $p_G = .3$. Thus $H_{adj} = -.7 \log_2 .7 - .3 \log_2 .3 = 0.88$ bits. The information content of this position is said to be $2 - 0.88 = 1.12$ bits. The more conserved the position, the higher the information content.

Exercise 1.4 Assume a model in which $p_{ij}$ is the probability of amino acid number $j$ occuring in the $i$th position of a sequence of length $l$. The amino acids are considered independent. What is the probability $P(a)$ of a particular sequence $a = a_1 \ldots a_l$? Show that the average of the log of the probability is the negative entropy

$$\sum_x P(x) \log P(x),$$

where the sum is over all possible sequences $x$ (of length $l$).

Coding theory and Shannon’s source coding theorem

The classical context for entropy is in coding theory, where it measures the number of binary events (bits) needed to transmit a signal. (In coding theory $\log_2$ is always assumed.)

Shannon’s source coding theorem states that, as $N \to \infty$, a message consisting of $N$ independent outcomes of the events $\{x_i\}$ can be transmitted with arbitrarily small error using a number of bits that tends to $N H(\{x_i\})$.

We can get an intuitive feel for this by considering the case where the $\{x_i\}$ consist of the two states ‘0’ and ‘1’, with probabilities $p_0$ and $p_1$, so a message consists of a string of $N$ ‘0’s or ‘1’s. Suppose $p_0 > p_1$. Then the probability of a message with $k$ ‘1’s is the term $\binom{N}{k} p_0^{N-k} p_1^k$ of the binomial distribution. This distribution has mean $\mu = N p_1$ and standard deviation $\sigma = \sqrt{N p_0 p_1}$. Almost all the probability of this distribution will lie in the range from $k = 0$ to $k = \mu + L \sigma$, where $L$ is set by the desired probability of error.

Now, the number of messages with $k$ ‘1’s is given by $\binom{N}{k}$ and this has its peak at $k = N/2$. We can use Stirling’s formula to approximate the value of $\binom{N}{k}$ at $k = N p_1$. Keeping only the predominant terms, we find

$$\binom{N}{N p_1} = \frac{N!}{(N p_1)! (N (1-p_1))!} \approx \frac{N^N e^{-N}}{(N p_1)^{N p_1} e^{-N p_1} (N (1-p_1))^{N (1-p_1)} e^{-N (1-p_1)}} = p_1^{-N p_1} p_0^{-N p_0} = 2^{-N H}$$

where $H$ is the entropy. To count the most probable messages, we want to sum $\binom{N}{k}$ from $k = 0$ to $k = \mu + L \sigma$, and it is easy to check that this increases the $N H$ bits to

$$\approx N (H + K / \sqrt{N})$$

for some constant $K$.

So, if we choose names for the $\approx 2^{NH}$ most probable messages, we can ensure that we are very unlikely to be presented with a message for which we have no name. As we can
construct the $2^NH$ names using only $NH$ bits, as opposed to the $N$ bits needed for all the messages, we have achieved a considerable data compression. Note, however, that when $p_0 = p_1 = \frac{1}{2}$ we have $H = \frac{1}{2} \log_2(2) + \frac{1}{2} \log_2(2) = 1$, so no data compression is achieved in this case; but this is obviously correct, since all messages then have the same probability.

We return to the definition of different types of entropy. The relative entropy (also known as the Kullback/Leibler “distance”) is defined by

$$H(p||q) = \sum_i p_i \log(p_i/q_i)$$

(1.10)

Information content and relative entropy are the same if the $q_i$ are an equiprobable ($q_i = \frac{1}{K}$) “background distribution” that represents a completely naive initial state for $H_{before}$. The two terms are sometimes used interchangeably.

Exercise 1.5 Prove the above assertion about the equivalence of information content and relative entropy.

Relative entropy has the property that it is always greater than or equal to zero. It is easy to show that $H(p||q) \geq 0$ with equality if and only if $p_i = q_i$ for all $i$ (see fig. 1.1).

The relative entropy often arises as the expected score in models where the score is defined as the log-odds, i.e. $P(data | M)/P(data | R)$, where $M$ is the model, and $R$ is a random, background model. If $p_i$ is the probability of the $i$-th residue in some position in a sequence according to $M$, and $q_i$ its probability according to $R$, then the score for residue $i$ is $\log(p_i/q_i)$, and the expected score is $\sum_i p_i \log(p_i/q_i)$, which is the relative entropy.

Another important entropy measure is the mutual information. If we have two sets of events, $X = \{x_i\}$ and $Y = \{y_j\}$, $\log\{P(x_i)P(y_j)/P(x_i,y_j)\}$ measures the information in the $(i,j)$-th event that is not due to the independent occurrence of $x_i$ and $y_j$. The mutual information, $H(X;Y)$, is the average of this, weighted by the probability $P(x_i,y_j)$:

$$H(X;Y) = \sum_{i,j} P(x_i,y_j) \log \frac{P(x_i,y_j)}{P(x_i)P(y_j)}$$

(1.11)

$H(X;Y)$ can be interpreted as the amount of information that acquire about outcome $X$ when we are told outcome $Y$.

Exercise 1.6 Show that $H(X;Y) = H(Y;X)$.

Exercise 1.7 Show that $H(X;Y) = H(X) + H(Y) - H(Y,X)$, where $H(Y,X)$ is the entropy of the joint distribution $P(X,Y)$. 

12
Figure 1.1: Proof that the relative entropy (1.10) is always positive or zero if \( p_i = q_i \) for all \( i \). This graph shows that \( \log(x) \leq x - 1 \) with equality only if \( x = 1 \). Then it follows that \(-H(p \mid q) = \sum_i p_i \log(q_i / p_i) \leq \sum_i p_i (q_i / p_i - 1) = 0\), with equality holding only if, for each \( i \), \( q_i = p_i \).

1.5 Inference

Probabilistic models are the main focus of this book. A model can be anything from a simple distribution to a complex stochastic grammar with many implicit probability distributions. Once the type of model is chosen, the parameters of the model have to be inferred from data. For instance, we may model the outcome of rolling a die with a multinomial distribution. Suppose the number of observations yielding \( i \) is \( n_i \) \((i = 1, \ldots, 6)\). We do not know if it is a fair die, so we need to estimate the parameters of the multinomial distribution, \( i.e. \), the probability \( \theta_i \) of getting \( i \) in a throw of the die.

Here, we consider the different strategies that might be used for inference in general. For more background, see (MacKay 1992).

Let us suppose, then, that we wish to infer parameters \( \theta = \{\theta_i\} \) for a model \( M \) from a set of data \( D \). The most obvious strategy is to maximise \( P(D \mid \theta, M) \) over all possible \( \theta \). This is called the maximum likelihood
criterion. Formally we write
\[ \theta^{\text{ML}} = \arg\max_{\theta} P(D \mid \theta, M). \] (1.12)

Generally speaking, when we treat \( P(x \mid y) \) as a function of \( x \) we refer to it as a probability; when we treat it as a function of \( y \) we call it a likelihood.

A drawback of maximum likelihood is that it can give poor results when the data are scanty; we would be wiser then to rely on more prior knowledge. Consider the dice example and assume we want to estimate the multinomial parameters from, say, 3 different rolls of the dice. It is shown in the next section that the maximum likelihood estimate of \( \theta_i \) is \( n_i / |n| \), i.e., it is 0 for at least three of the parameters. This is obviously a bad estimate for most dice, and we would like a way to incorporate the prior knowledge that we expect all the parameters to be quite close to \( 1/6 \).

The way to introduce prior knowledge is to use Bayes’ theorem. Suppose there is a probability distribution over the parameters \( \theta \). Conditioning throughout on \( M \) gives the following version of Bayes’ theorem:
\[ P(\theta \mid D, M) = \frac{P(D \mid \theta, M)P(\theta \mid M)}{P(D \mid M)}. \] (1.13)

The prior \( P(\theta \mid M) \) has to be chosen in some reasonable manner, and that is the art of Bayesian estimation. This freedom to choose a prior has made Bayesian statistics controversial at times, but we believe it is a very convenient framework for incorporating prior (biological) knowledge into statistical estimation.

Given Bayes’ theorem, there are two estimation criteria to choose from. The obvious generalization of ML is the maximum a posteriori probability (MAP) estimate,
\[ \theta^{\text{MAP}} = \arg\max_{\theta} P(D \mid \theta, M)P(\theta \mid M). \] (1.14)

Note that we ignore the data prior \( P(D \mid M) \), because it does not depend on the parameters \( \theta \) and thus the maximum point \( \theta^{\text{MAP}} \) is independent of it.

The MAP estimator is considered a little suspicious, because we are maximising a probability density; a non-linear transformation of the parameters usually changes the result. In technical terms it is not invariant (see below). The ML estimator does not suffer from this problem, and neither
does the \textit{posterior mean} estimator (PME), which chooses the average of all parameter sets weighted by the posterior:

\[ \theta^{\text{PME}} = \int \theta P(\theta | n) d\theta. \]

(1.15)

The integral is over all valid probability vectors, \textit{i.e.}, all those that sum to one. In the following we will derive the PME for a multinomial distribution with a certain prior.\(^1\)

\section*{Invariance}

Given a density \( f(y) \), suppose there is a change of variable \( y = \phi(x) \). Then we can define a density \( g(x) \) by \( g(x) = f(\phi(x))\phi'(x) \). The derivative of \( \phi \), \( \phi'(x) \), is there because the interval \( \delta y \) corresponds to an interval \( \delta x \phi'(x) \) under the transform \( \phi \), so the amount of the \( f \) density that is swept out under \( \phi \) is proportional to this derivative.

This definition produces a correctly normalised density because

\[ \int g(x) dx = \int f(\phi(x))\phi'(x) dx = \int f(y) dy = 1, \] since \( f \) is a density. We write the transformation rule formally as

\[ g(x) = f(\phi(x))/\phi'(x). \]

Note that densities transform \textit{contravariantly}, in the sense that \( y = \phi(x) \) is a mapping from the \( x \)-space to the \( y \)-space, but the density \( f \) on the \( y \)-space is transformed to a density on the \( x \)-space; densities are mapped in the opposite direction from the underlying map of the spaces.

In particular the maximum point of a density can change under a non-linear transformation. The MAP estimate is not invariant for that reason.

The likelihood is not a probability density, and therefore is not rescaled under a change of variables. Its maximum is therefore independent of the choice of variables, and the maximum likelihood estimate is invariant.

\section*{1.6 Estimation of Probabilities From Counts}

Above we used an example of rolling a die. We needed to estimate the parameters of a multinomial from data, \textit{i.e.}, examples of throws. The same abstract situation occur frequently in sequence analysis, but with the number of rolls \( n_i \) with outcome \( i \) now meaning something different. For instance, \( n_i \) might be the number of times amino acid \( i \) occurs in a column of a multiple alignment.

\(^1\) Having noted the problem with MAP, we should also say that the posterior mean estimator sometimes gives useless results. This, for instance, is the case when there are two maxima of the posterior (\textit{i.e.}, two equally good MAP solutions) because of some symmetry in the problem. In such a case an average may give, say, a uniform set of parameters even if the two MAP solutions are far from uniform.
Assume that the observations can be expressed as counts $n_i$ for outcome $i$, $i = 1, \ldots, K$. If we have plenty of data, it is natural to use the observed frequencies, $\hat{\theta}_i = n_i / n$, as the estimated probabilities. Here $|n| = \sum_i n_i$. This is the maximum likelihood solution, $\theta_{\text{ML}}$. The proof that this is so goes as follows.

We want to show that $P(n|\theta_{\text{ML}}) > P(n|\theta)$ for any $\theta \neq \theta_{\text{ML}}$. This is equivalent to showing that $\log[P(n|\theta_{\text{ML}})/P(n|\theta)] > 0$, if we only consider probability parameters yielding a non-zero probability. Using equations (1.4) and the definition of $\theta_{\text{ML}}$, this becomes

$$\log \frac{P(n|\theta_{\text{ML}})}{P(n|\theta)} = \log \frac{\prod (\theta_i^{ML})^{n_i}}{\prod \theta_i^{n_i}} = \sum_i n_i \log \frac{\theta_i^{ML}}{\theta_i} = |n| \sum_i \theta_i^{ML} \log \frac{\theta_i^{ML}}{\theta_i} > 0.$$  

The last inequality follows from the fact that the relative entropy is always positive except when the two distributions are identical. This concludes the proof.\footnote{Readers familiar with Lagrange multipliers may find it easier to prove the ML formula by first differentiating $P(n|\theta)$ under the constraint that the $\theta$s sum to 1 and then setting the derivatives equal to zero and solving for $\theta$.}

If data are scarce, it is not so clear what is the best estimate. If, for instance, you only have a total of two counts both on the same residue, the maximum likelihood estimate would give zero probability to all other residues. In this case, one would like to assign some probability to the other residues and not rely entirely on so few observations. Since there are no more observations, these probabilities must be determined from prior knowledge. This can be done via Bayesian statistics, and we will now derive the posterior mean estimator for $\theta$.

As the prior we choose the Dirichlet distribution (1.6) with parameters $\alpha$. We can then calculate the posterior (1.13) for the multinomial distribution with observations $n$:

$$P(\theta|n) = \frac{P(n|\theta)D(\theta|\alpha)}{P(n)}.$$  

For ease of notation, we have dropped the conditioning on the model $M$ as compared to (1.13), and consider all probabilities implicitly conditioned on
the model. Inserting the multinomial distribution (1.4) for $P(n|\theta)$ and the expression (1.6) for $D(\theta | \alpha)$ yields

$$P(\theta|n) = \frac{1}{P(n)Z(\alpha)M(n)} \prod_i^{\alpha_i} \theta_i^{n_i} = \frac{Z(n + \alpha)}{P(n)Z(\alpha)M(n)} D(\theta | n + \alpha).$$

(1.16)

In the last step $\prod_i^{\alpha_i} \theta_i^{n_i}$ was recognized as being proportional to the Dirichlet distribution with parameters $n + \alpha$. Fortunately we do not have to get involved with gamma functions in order to finish the calculation, because we know that both $P(\theta|n)$ and $D(\theta | n + \alpha)$ are properly normalized probability distributions over $\theta$. This means that all the prefactors must cancel and

$$P(\theta|n) = D(\theta | n + \alpha).$$

(1.17)

We see that the posterior is itself a Dirichlet distribution like the prior, but of course with different parameters.

Now, we only need to perform an integral in order to find the posterior mean estimator. From the definition (1.15),

$$\theta_i^{\text{FME}} = \int \theta_i D(\theta | n + \alpha) d\theta = Z^{-1}(n + \alpha) \int \theta_i \prod_k \theta_k^{n_k + \alpha_k} d\theta.$$  

(1.18)

We can bring $\theta_i$ inside the product giving $\theta_i^{n_i + \alpha_i}$ as the $i$th term. Then we see that the integral is exactly of the form (1.7). We can therefore write

$$\theta_i^{\text{FME}} = \frac{Z(n + \alpha + \delta_i)}{Z(n + \alpha)} = \frac{n_i + \alpha_i}{|n| + |\alpha|},$$

(1.19)

where $\delta_i$ is a vector whose $i$th component is 1 and all its other components zero. Here we have used the property (1.8) of the gamma function, i.e. $\Gamma(x + 1) = x \Gamma(x)$; this allows us to cancel all terms except $n_i + \alpha_i$ in the numerator and $|n| + |\alpha|$ in the denominator.

This result should be compared to the ML estimate $\theta_{\text{ML}}$. If we think of the $\alpha$’s as extra observations added to the real ones, this is precisely the ML estimate! The $\alpha$’s are like pseudocounts added to the real counts. This makes the Dirichlet regularizer very intuitive, and one can in a sense forget all about Bayesian statistics and think in terms of pseudocounts. It is fairly obvious how to use these pseudocounts: If it is known a priori that a certain residue, say number $i$, is very common, one should give it a high
pseudo count \( \alpha_i \), and if residue \( j \) is generally rare, one should give it a low pseudo count.

It is important to note the self-regulating property of the pseudo count regularizer: If there are many observations, i.e., the \( n \)'s are much larger than the \( \alpha \)'s, then the estimate is essentially equal to the ML estimate. On the other hand, if there are very few observations, the regularizer would dominate and give an estimate close to the normalized \( \alpha \)'s, \( \theta_i \approx \alpha_i / |\alpha| \).

So typically one would choose the \( \alpha \)'s so that they are equal to the overall distribution of residues after normalization.

If it were not for the intuitive beauty of formula (1.19), Dirichlet priors might not have been so widely used. However, the statistical framework is not only elegant, but highly advantageous, for it allows one to estimate the parameters for the Dirichlet distribution from real biological data, as will be explained later.

### 1.7 Mixtures of Dirichlets

It is not easy to express all the prior knowledge about proteins in a single Dirichlet distribution; to achieve that it is natural to use several different Dirichlet distributions. One might for instance have a Dirichlet well suited for exposed amino acids, one for buried ones and so forth. In statistical terms this can be expressed as a mixture distribution. Assume we have \( m \) Dirichlet distributions characterized by parameter vectors \( \alpha^1 \ldots \alpha^m \). A mixture prior expresses the idea that any probability vector \( \theta \) belongs to one of the components of the mixture \( \mathcal{D}(\theta | \alpha^k) \) with a probability \( q_k \). Formally:

\[
P(\theta | \alpha^1 \ldots \alpha^m) = \sum_k q_k \mathcal{D}(\theta | \alpha^k),
\]

where \( q_k \) are called the mixture coefficients. The mixture coefficients have to be positive and sum to one in order for the mixture to be a proper probability distribution. (Mixtures can be formed from any types of distributions in this way.) Whereas this probability was called \( P(\theta) \) in the previous section, we are here conditioning on the \( \alpha \)'s, which was implicit before. This turns out to be convenient, because we can then use probabilities like \( P(\alpha^1 | \mathbf{n}) \) (see below). We can then also identify \( q_k \) as the prior probability \( q_k = P(\alpha^k) \) of each of the mixture coefficients.

For a given mixture, i.e., for fixed \( \alpha \) parameters and mixture coefficients, it is straightforward to calculate the posterior probabilities using the results
from the previous section. From the definition of conditional probabilities, and (1.19) we have

\[
P(\theta \mid n) = \sum_k P(\theta \mid \alpha^k, n) P(\alpha^k \mid n) = \sum_k P(\alpha^k \mid n) D(\theta \mid n + \alpha^k),
\]

where we used the expression for the posterior (1.17). To compute the term \(P(\alpha^k \mid n)\), note that by Bayes’ theorem we have

\[
P(\alpha^k \mid n) = \frac{q_k P(n \mid \alpha^k)}{\sum_l q_l P(n \mid \alpha^l)},
\]

using the fact that \(q_k = P(\alpha^k)\). Then

\[
P(n \mid \alpha^k) = \int P(n, \theta \mid \alpha^k) d\theta = \int P(n \mid \theta, \alpha^k) P(\theta \mid \alpha^k) d\theta = \frac{Z(n + \alpha^k)}{Z(\alpha^k) M(n)}
\]

Notice that we could have derived this from the fact that the coefficient in (1.16) is 1. This implies that \(P(n) = \frac{Z(n + \alpha)}{Z(\alpha) M(||n||)}\), and, given that we are conditioning on \(\alpha^k\) here, the result follows.

The final integration to obtain \(\theta_{\text{FME}}\) can be done using the results from the previous section (1.18) and (1.19), and yields

\[
\theta_{\text{FME}} = \sum_k P(\alpha^k \mid n) \frac{n_i + \alpha^k_i}{n + |\alpha^k|},
\]

where

\[
P(\alpha^k \mid n) = \frac{Z(n + \alpha^k)/Z(\alpha^k)}{\sum Z(n + \alpha^l)/Z(\alpha^l)}.
\]

### 1.8 Estimating the Prior

For more details of the ideas presented in the preceding section, see (Brown et al. 1993) and (Sjölander et al. 1996). These authors used Dirichlet mixtures to model the distribution of column counts. They obtained the
prior by estimating the mixture components and the mixture coefficients from a large data set, i.e. a large set of count vectors.

The estimation is done as follows: The mixture defines a probability for each count vector in the data base, \( n^1, \ldots, n^M \),

\[
P(n^1 | \alpha^1 \ldots \alpha^m, q_1 \ldots q_m) = \int P(n^1 | \theta) P(\theta | \alpha^1 \ldots \alpha^m, q_1 \ldots q_m) d\theta. \quad (1.22)
\]

If the count vectors are considered independent the total likelihood of the mixture is

\[
P(\text{data} | \text{mixture}) = \prod_{i=1}^{M} P(n^i | \alpha^1 \ldots \alpha^m, q_1 \ldots q_m). \quad (1.23)
\]

This probability can be maximized by gradient descent or some other method of continuous optimization.

At this point the reader is probably asking “why use ML estimation instead of these wonderful Bayesian approaches I just learned.” To do this you just need a prior on the parameters of the first level of priors. You can put priors on prior parameters forever. At some point you have to settle for a prior you invented or one estimated by ML or some other non-Bayesian method.

### 1.9 Sampling

Given probabilities \( P(a_i) \) defined on the members \( a_i \) of a finite set, such as the set of amino acids, to **sample** from this set means to pick elements \( a_i \) randomly with probability \( P(a_i) \). In practice, sampling is accomplished by using pseudo-random numbers produced by the `rand()` function (or something similar) on a computer. If \( x = \text{rand()} \) picks numbers randomly and uniformly from the interval \([0, 1]\), then we can choose our element \( a_i \) by finding that \( i \) for which \( P(a_1) + \ldots + P(a_{i-1}) < \text{rand()} < P(a_1) + \ldots + P(a_i) \). Clearly, the probability of \( \text{rand()} \) lying in this range is \( P(a_i) \), so \( a_i \) is picked with the correct probability.

The concept of sampling applies also to densities: Given a density \( f \), to sample from it is to pick elements \( x \) from the space on which \( f \) is defined so that the probability of picking a point in an arbitrarily small region \( \delta R \) round the point \( x \) is \( f(x) \delta R \). Sampling of densities can also be accomplished by using pseudo-random numbers. These can be regarded as sampling from the
uniform density on \([0, 1]\), and if we wish to sample from some other density, we can apply a change of variables that changes the density appropriately.

The theory of this goes as follows: Suppose we are given a density \(f(x)\), and a map \(y = \phi(x)\). If we sample on the \(x\)-space and map each sample point \(x\) to \(\phi(x)\) on the \(y\)-space, the more rapidly the map \(\phi(x)\) changes with respect to \(x\), the more the sample points get spread out, and hence the smaller the density induced on the \(y\)-space. More precisely, the density \(g(y)\) induced by \(\phi\) is given by \(g(y) = f(x) \frac{dx}{dy} = f(x)/\phi'(x)\). (In the case where there are several variables, this derivative has to be replaced by a Jacobian matrix, which measures the size of a volume element in the \(y\)-space. Otherwise the rule is the same.)

Suppose for instance that we want to sample from a Gaussian. We define the cumulative Gaussian map \(\psi(y) = \int_0^y e^{-u^2/2}/\sqrt{2\pi} du\), and let \(y = \psi^{-1}(x)\), where \(\psi^{-1}\) is the inverse function of \(\psi\). Then \(x = \psi(y)\) so \(\frac{dx}{dy} = e^{-\psi^2/2}/\sqrt{2\pi}\). If \(f\) is the flat density on \([0, 1]\), \(f(x) = 1\) for all \(x\). So the density \(g\) induced by \(\psi^{-1}\) is \(\frac{dg}{dy} = e^{-\psi^2/2}/\sqrt{2\pi}\), which is the required Gaussian. Most computers don’t have a readily available inverse cumulative Gaussian function, so some other approach may be more convenient. For instance, one can pick two random numbers \(x\) and \(y\) in the range \([0, 1]\) and map the pair \((x, y)\) to the sample point \(\cos(2\pi x)\log(1/y^2)\).

**Exercise 1.8** (Calculus needed!) Prove that this last method samples correctly from a Gaussian.

We sometimes want to sample sets of probabilities. Suppose, for instance, that we want to choose sets of probabilities \((p_1 \ldots p_{20})\) for the 20 amino acids in some hypothetical environment. We could try choosing 20 numbers \(x_1 \ldots x_{20}\), generating each by \(\text{rand()}\), and then normalising their sum to give probabilities \(p_i = x_i/\sum_j x_j\). However, these would not give a uniform distribution on the space of numbers \((p_1 \ldots p_{20})\) satisfying \(\sum_i p_i = 1\); instead, they would tend to concentrate around the region where all the \(p_i\) are equal. The following algorithm corrects this defect:

\[
\begin{align*}
s &= 0 \\
&\text{for } i = 1 \text{ to } 19 \\
&\quad \{ \\
&\quad \quad p_i = (1 - s)(1 - \text{rand()}^{1/(20-i)}) \\
&\quad \quad s = s + p_i \\
&\quad \} \\
&\quad p_{20} = 1 - s
\end{align*}
\]
Exercise 1.9 Prove this algorithm samples from the probability space correctly.

Often we want to sample not from some well-defined analytic function, like \( e^{-x^2/2} \), but from a probabilistic model of many variables where we can compute the density at each point, but where it would be impractical to evaluate derivatives or to integrate the probability over all the variables. Then we need an alternative approach to sampling, and a widely used method is Gibbs' sampling. This works by choosing points from the conditional distribution \( P(x_i \mid x_1 = a_1 \ldots x_{i-1} = a_{i-1}, x_{i+1} = a_{i+1} \ldots x_N) \) for each \( i \), cycling repeatedly through \( i = 1 \ldots N \). It is generally much easier to sample from this distribution of one variable than from the whole \( N \)-dimensional space.

The proof that this truly samples correctly from the whole space will not be given here, but the idea in outline is to show that, if we condition on variable \( i \) and if \( (a_1 \ldots a_N) \) has been sampled with the probability \( P(a_1 \ldots a_N) \), then points \( (a_1 \ldots a_{i-1}, x_i, a_{i+1} \ldots a_N) \) will be sampled with probability \( P \) by the above conditional distribution. Thus once the Gibbs' sampling process tends towards picking points with density \( P \) it will continue doing so. Provide that the process doesn't get stuck in some subset of the parameter space (provided it is 'ergodic'), it can be shown it will inevitably converge to \( P \).

The kind of situation in which Gibbs' sampling can get stuck is where there are two pieces of density which do not overlap along any of the coordinate directions, eg in the 2D case where half the density lies in the region \([0,1] \times [0,1]\) and the other half in the region \([2,3] \times [2,3]\). Note that if there were even a small overlap, eg if half the density were uniform on \([0,1] \times [0,1]\) and the other half uniform on \([.99,1.99] \times [.99,1.99]\), then sampling would pass between the two regions, albeit making the transition between regions quite infrequently.

Exercise 1.10 What is the expected number of samples within one region, in the preceding example, before a cross-over occurs into the other?

1.10 Extreme Value Distributions

Suppose one takes \( N \) samples from the distribution \( g(x) \). The probability that the largest amongst them is less than \( x \) is \( G(x)^N \), where \( G(x) = \int_{-\infty}^{x} g(u) du \). The distribution of the largest of the set of \( N \) is given by differentiating this with respect to \( x \), giving \( Ng(x)G(x)^{N-1} \). The limit for large \( N \) of \( G(x)^N \) is called the extreme value distribution for \( g(x) \). It
can be used for modeling the distribution of maximal scores when a given
sequences is tested against a database.

Let’s compute the extreme value distribution when \( g(x) \) is the exponential \( g(x) = \alpha e^{-\alpha x} \) and \( G(x) = 1 - e^{-\alpha x} \). Choosing \( y \) so that \( e^{-\alpha y} = 1/N \) we find \( G(x)^N = (1 - e^{-\alpha x})^N = (1 - e^{-\alpha(x-y)})^N = (1 - e^{-\alpha(x-y)/N})^N \). For large \( N \), \( G(x)^N \simeq \exp(-e^{-\alpha(x-y)}) \). This exponential-of-an-exponential is called a Gumbel distribution.

Consider next the case where \( g(x) \) is the Gaussian \( e^{-x^2/2} \). Putting \( v = u - y \) in the integral \( \int_y^x e^{-u^2/2} \, du \) gives

\[
\int_y^x e^{-u^2/2} \, du = \int_0^{x-y} e^{-(u+y)^2/2} \, dv = e^{-y^2/2} \int_0^{x-y} e^{-vy} e^{-v^2/2} \, dv
\]

If \( y \) is large, \( e^{-v^2/2} \simeq 1 \) for those values of \( v \) for which \( e^{-vy} \) makes significant contributions to the integral. So we can ignore the term \( e^{-v^2/2} \), giving

\[
\int_y^x e^{-u^2/2} \, du \simeq \frac{e^{-y^2/2}}{y} (1 - e^{-y(x-y)})
\]

Putting \( x = \infty \) gives \( \int_y^\infty e^{-u^2/2} \simeq e^{-y^2/2}/y \). Choose \( y \) such that \( e^{-y^2/2}/y = 1/N \) (so \( y \simeq \sqrt{2 \log N} \)). Then \( \int_x^\infty e^{-u^2/2} = -\int_y^x e^{-u^2/2} + \int_y^\infty e^{-u^2/2} = e^{-y(x-y)}/N \). So \( G(x)^N = (1 - e^{-y(x-y)/N})^N \). For large \( N \), \( G(x)^N \simeq \exp(-e^{-y(x-y)}) \).

Once again we get a Gumbel distribution, which should tip us off that something interesting is going on. In fact, one can show that an extreme value distribution must have the form \( \exp(-f(a_N x + b_N)) \), where \( a_N \) and \( b_N \) are constants depending on \( N \) and \( f(x) \) is either an exponential \( e^{-x} \) or \( |x|^{-\lambda} \) for some positive constant \( \lambda \) (see (Waterman 1995) for a more precise statement of this theorem). For both the exponential and Gaussian distributions, \( f \) is exponential, as we have seen.
References


